

XMC1200 Boot Kit

Getting Started



Agenda (1/2)

1

Kit Overview

2

Hardware Overview

3

Tooling Overview

4

– Boot Modes

5

– DAVE™

6

Getting Started

7

– Example – Blinky based on XMC Lib

8

– Example – Blinky based on DAVE™ APPs

Agenda (1/2)

1

Kit Overview

2

Hardware Overview

3

Tooling Overview

4

– Boot Modes

5

– DAVE™

6

Getting Started

7

– Example – Blinky based on XMC Lib

8

– Example – Blinky based on DAVE™ APPs

Agenda (1/2)

1

Kit Overview

2

Hardware Overview

3

Tooling Overview

4

– Boot Modes

5

– DAVE™

6

Getting Started

7

– Example – Blinky based on XMC Lib

8

– Example – Blinky based on DAVE™ APPs

Agenda (1/2)

1

Kit Overview

2

Hardware Overview

3

Tooling Overview

4

– Boot Modes

5

– DAVE™

6

Getting Started

7

– Example – Blinky based on XMC Lib

8

– Example – Blinky based on DAVE™ APPs

Agenda (1/2)

1

Kit Overview

2

Hardware Overview

3

Tooling Overview

4

– Boot Modes

5

– DAVE™

6

Getting Started

7

– Example – Blinky based on XMC Lib

8

– Example – Blinky based on DAVE™ APPs

Agenda (1/2)

1

Kit Overview

2

Hardware Overview

3

Tooling Overview

4

– Boot Modes

5

– **DAVE™**

6

Getting Started

7

– Example – Blinky based on XMC Lib

8

– Example – Blinky based on DAVE™ APPs

Agenda (1/2)

1

Kit Overview

2

Hardware Overview

3

Tooling Overview

4

– Boot Modes

5

– DAVE™

6

Getting Started

7

– Example – Blinky based on XMC Lib

8

– Example – Blinky based on DAVE™ APPs

Agenda (1/2)

1

Kit Overview

2

Hardware Overview

3

Tooling Overview

4

– Boot Modes

5

– DAVE™

6

Getting Started

7

– Example – Blinky based on XMC Lib

8

– Example – Blinky based on DAVE™ APPs

Agenda (1/2)

1

Kit Overview

2

Hardware Overview

3

Tooling Overview

4

– Boot Modes

5

– DAVE™

6

Getting Started

7

– Example – Blinky based on XMC Lib

8

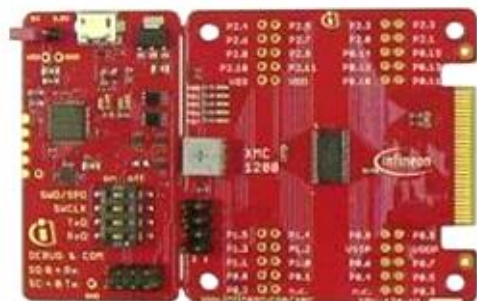
– Example – Blinky based on DAVE™ APPs

Kit Overview (1/2)

> XMC1200 Boot Kit

- Consists of an XMC1200 CPU Card
- Supported Application Card examples: Colour LED Card, White LED Card

(Application Cards are orderable separately or as part of another Application Kit)



XMC1200 CPU Card



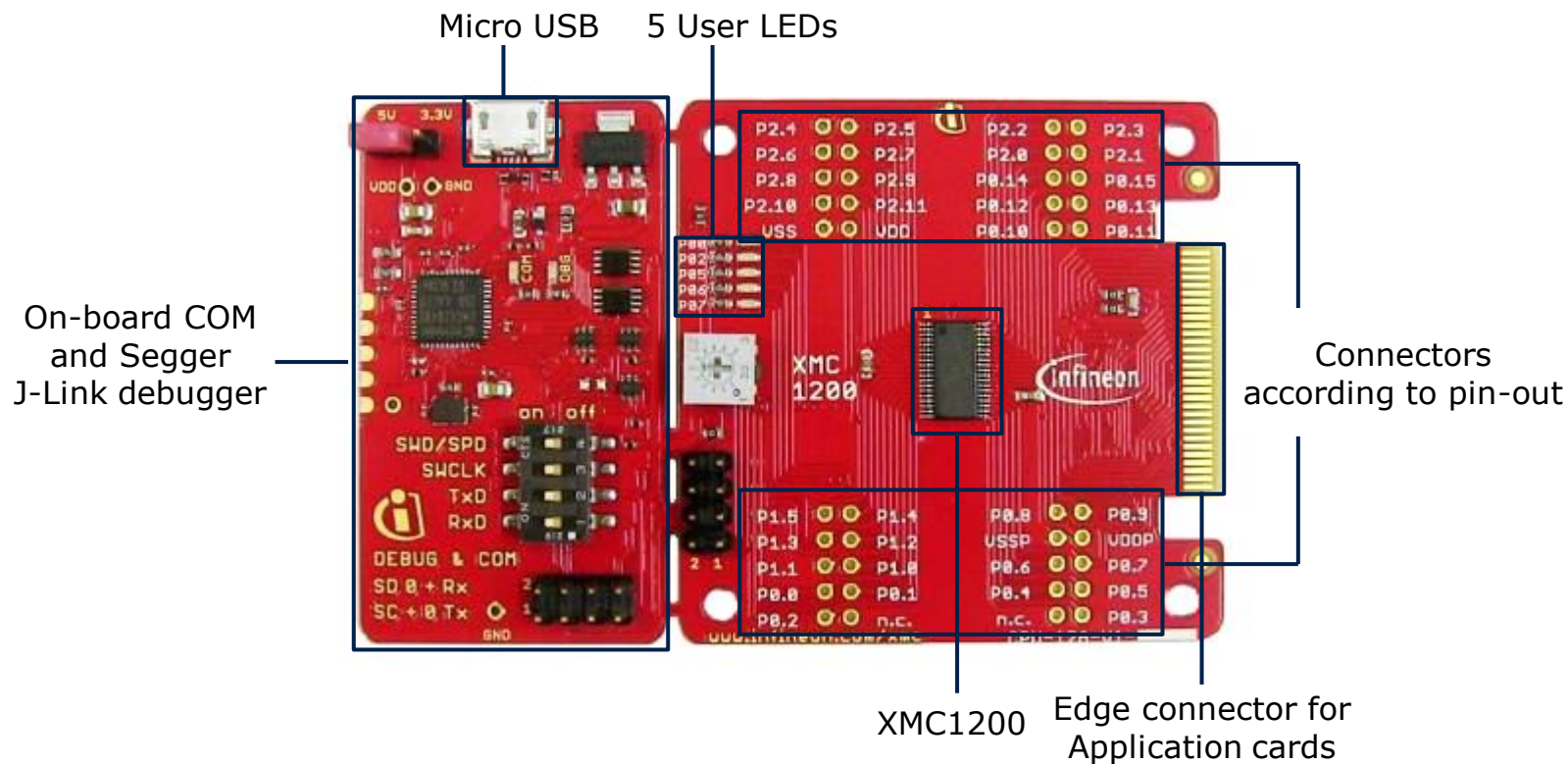
Colour LED Card



White LED Card

Kit Overview (2/2)

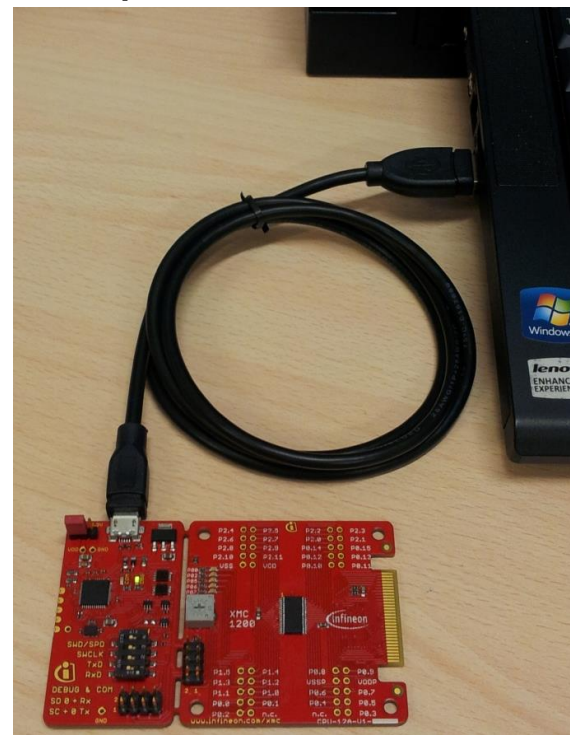
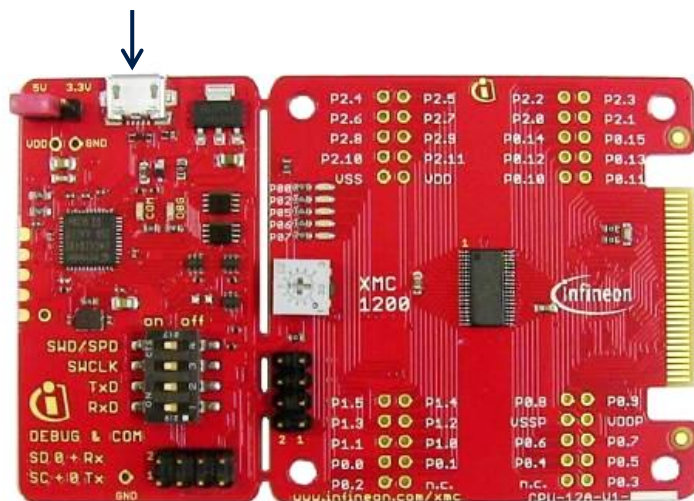
> XMC1200 CPU Card



Hardware Overview

- › Connect XMC1200 CPU Card to PC via USB cable
- › CPU Card is powered up (as indicated by LED on the card)

CPU Card powered via USB cable



- › Note: Supported Application Card may be additionally connected to the CPU card

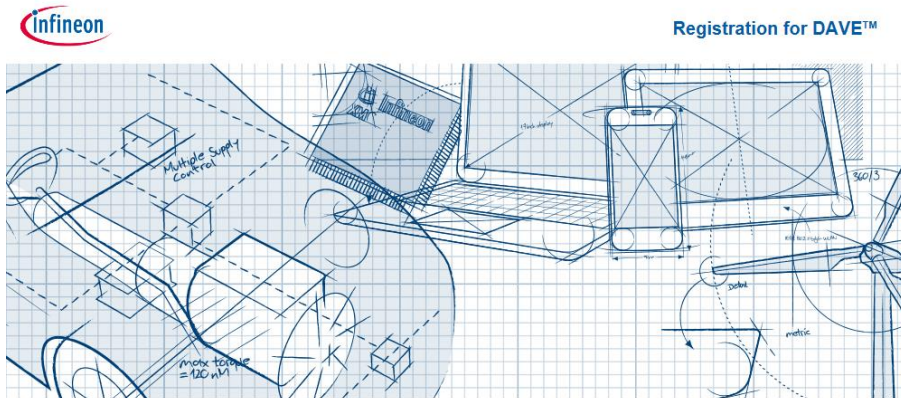
- › Boot Modes available
 - UART Bootstrap-Loader Mode
 - User Mode (Halt After Reset)
 - User Mode (Debug) **Default Mode of device on Boot Kit**
 - User Mode (Productive)

- › Boot Modes can be configured via:
 - DAVE™
 - Download DAVE™
[DAVE™ v4.1.2 download](#)
 - MemTool
 - Download MemTool
[MemTool v4.65.exe download](#)

- › For more information on how to configure the BMI value, please refer to the XMC1000 Tooling Guide.

Tooling Overview – DAVE™ (1/5)

- › DAVE™ download package is available at:
<http://infineon-community.com/LP=400>



Please register to download DAVE™ version 4 and DAVE™ SDK version 4.

DAVE™ version 4 and DAVE SDK version 4 is now available as productive version.
The current versions are: DAVE™ v4.1.2 and DAVE™ SDK v4.1.2.

After registration you will receive a confirmation email with a link to the download-page. With a click on the link you can download a zip file that contains a setup.exe-file and a PDF-file with installation instructions.
Please check the JUNK or SPAM folder of your mail server if you don't receive a confirmation email.



First Name*	<input type="text"/>
Last Name*	<input type="text"/>
Email Address*	<input type="text"/>
Country*	<input type="text" value="-- Please select --"/>
Company*	<input type="text"/>
Business Phone	<input type="text"/>
Target Application	<input type="text" value="--please select--"/>

Tooling Overview – DAVE™ (2/5)

› After registration, download and unzip the installer package

› Run DAVE-4.1.2-Setup.exe to install
DAVE™ IDE and SEGGER J-Link drivers

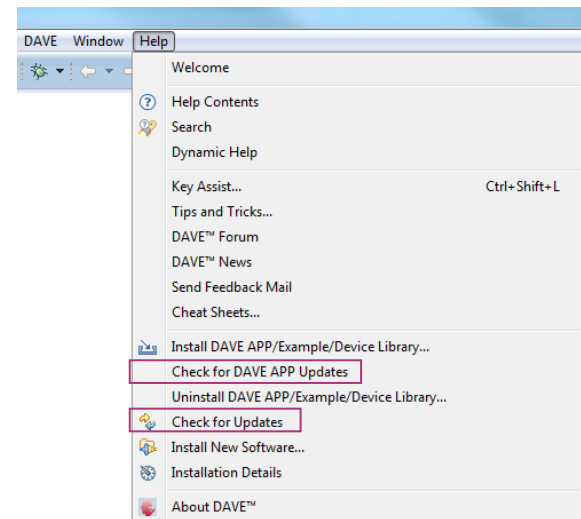
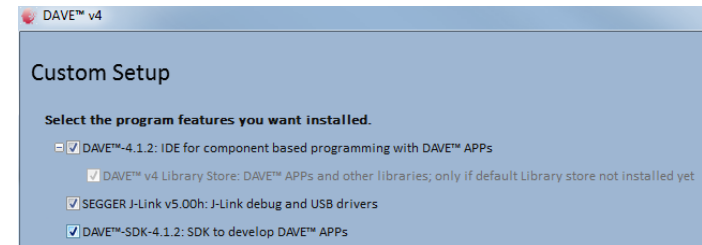
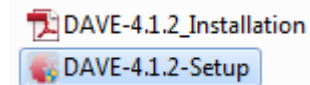
› Open DAVE™



› Update DAVE™ and DAVE™ libraries

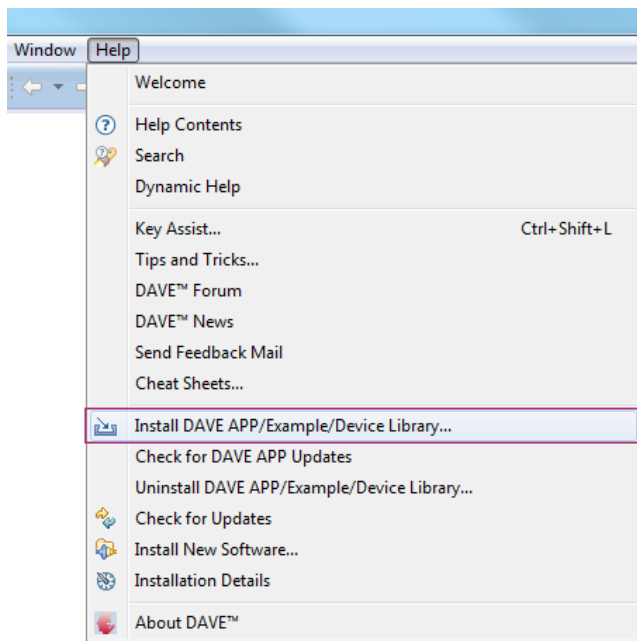
– Help → Check for Updates

– Help → Check for DAVE APP Updates



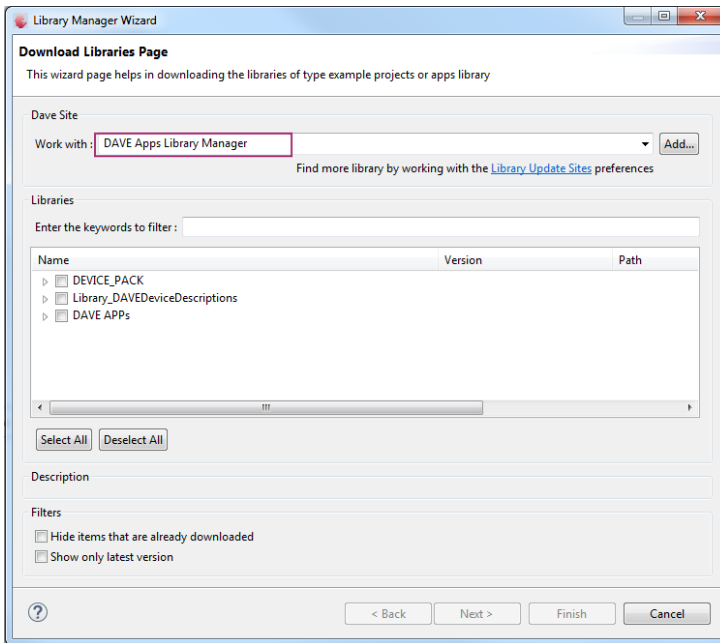
Tooling Overview – DAVE™ (3/5)

- › Install DAVE™ APPs libraries and Device Description
 - Help → Install DAVE APP/Example/Device Library



- › Note: You may skip the above step if you are not using DAVE™ APPs

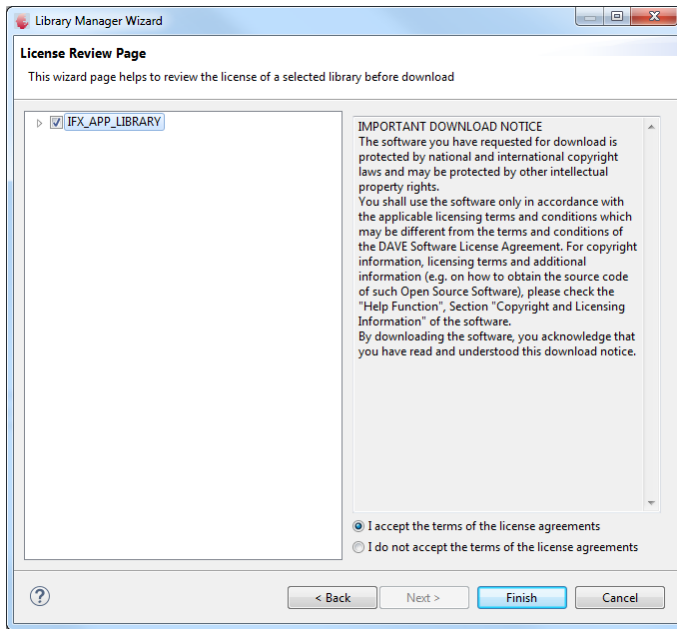
- › Select DAVE Apps Library Manager in the drop-down menu



- › Select DEVICE_PACK, Library_DAVEDeviceDescriptions (XMC1200 Device) and DAVE APPs

- ▶ DEVICE_PACK
- ▶ Library_DAVEDeviceDescriptions
- ▶ DAVE APPs

- › Accept terms of the license agreements and click Finish



- › DAVE™ APPs libraries and Device Description are installed

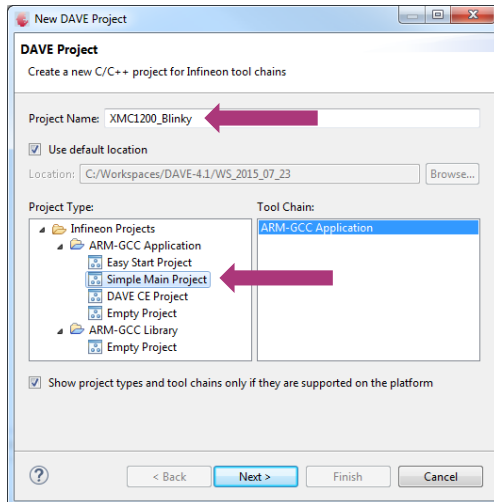
Getting Started – Example – Blinky based on XMC Lib (1/6)

1. Open DAVE™

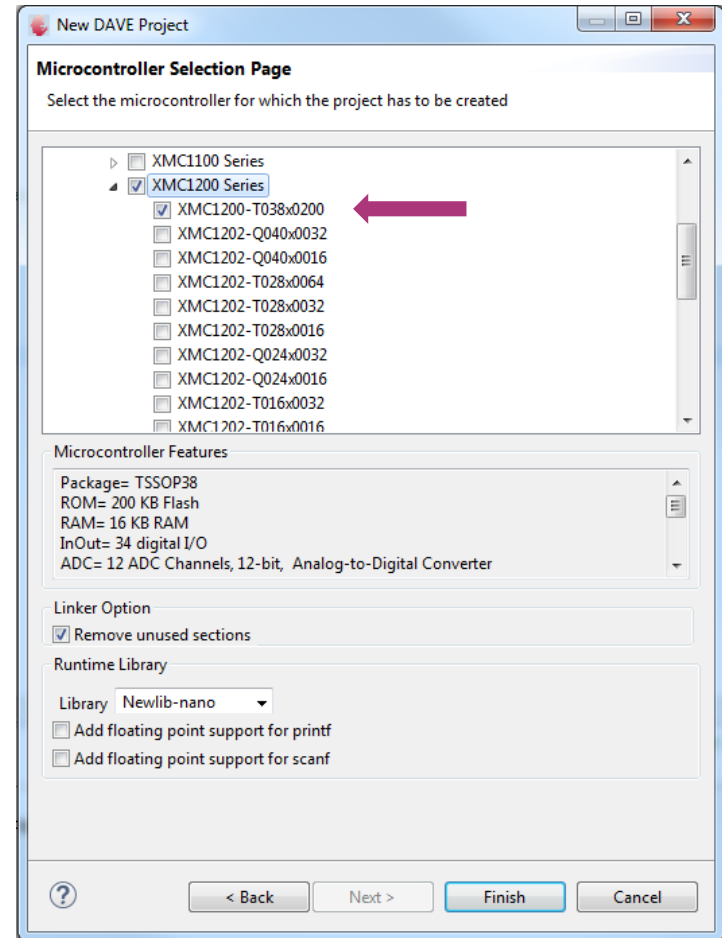


2. Create a new "Simple Main" project:

- File → New → DAVE Project
- Enter project name e.g. "XMC1200_Blinky"
- Select "Simple Main Project" as Project Type



3. Select the device accordingly



Getting Started – Example – Blinky based on XMC Lib (2/6)



- › For this project, we will use
 - System clock frequency of 8MHz
 - LED on Port pin 0.0
 - System timer, SysTick, as the time base for interrupt

- › Next, we will show you how to
 1. Set up the System or Main Clock (MCLK)
 2. Configure Port pin
 3. Configure SysTick and define its exception service routine

Getting Started – Example – Blinky based on XMC Lib (3/6)



1. Set up System or Main Clock (MCLK) using XMCLib

- Include the header files required for MCLK and GPIO configuration

```
#include "xmc_gpio.h"  
#include "xmc_scu.h"
```

- MCLK configured via **IDIV** and **FDIV** bit fields in XMC_SCU_CLOCK_CONFIG data structure

```
XMC_SCU_CLOCK_CONFIG_t clock_config =  
{  
    .pclk_src = XMC_SCU_CLOCK_PCLKSRC_DOUBLE_MCLK, /*PCLK = 2*MCLK*/  
    .rtc_src = XMC_SCU_CLOCK_RTCCLKSRC_DCO2,  
    .fdiv = 0, /**< Fractional divider */  
    .idiv = 4, /**MCLK = 8MHz */  
};
```

- Initializes clock generators and clock tree in **Main.c**

```
int main(void)  
{  
  
    /* Ensure clock frequency is set at 8MHz (MCLK) */  
    XMC_SCU_CLOCK_Init(&clock_config);  
}
```

2. Configure Port pin

- GPIO to toggle the LED is configured via **mode** and **output_level** of XMC_GPIO_CONFIG structure.

```
XMC_GPIO_CONFIG_t gpio_output_config =  
{  
    .mode           = XMC_GPIO_MODE_OUTPUT_PUSH_PULL,  
    .output_level   = XMC_GPIO_OUTPUT_LEVEL_HIGH,  
};
```

- Initializes port pin P0.0 as general purpose output pin in **Main.c**

```
/* Initialise P0.0 as an output pin */  
XMC_GPIO_Init(LED1, &gpio_output_config);
```

3. Configure SysTick and define its exception service routine

- SysTick exception handler is defined in **startup_XMC1200.s**

```
/* ===== */
    .globl SysTick_Veneer
SysTick_Veneer:
    LDR R0, =SysTick_Handler
    MOV PC,R0
/* ===== */
```

- Initialize the SysTick in **Main.c**

```
/* System timer configuration */
    SysTick_Config(SystemCoreClock / TICKS_PER_SECOND);
```

- Define the SysTick exception handler routine in Main.c

```
void SysTick_Handler(void)
{
    static uint32_t ticks = 0;

    ticks++;
    if (ticks == TICKS_WAIT)
    {
        XMC_GPIO_ToggleOutput(LED1);
        ticks = 0;
    }
}
```




Getting Started – Example – Blinky based on XMC Lib (6/6)

› Build project

1. Click 
2. Wait for Build to finish

```
'Invoking: ARM-GCC Print Size'  
"C:\DAVEv4\DAVE-4.1.2\eclipse\ARM-GCC-49\bin/arm-none-eabi-  
text  data  bss  dec  hex filename  
2232   20  1040  3292   cdc XMC1200_Blinky.elf  
'Finished building: XMC1200_Blinky.siz'
```

› Download code

1. Click 
2. Switch to Debug perspective
3. Click  to run code



› LED blinks every 0.2s



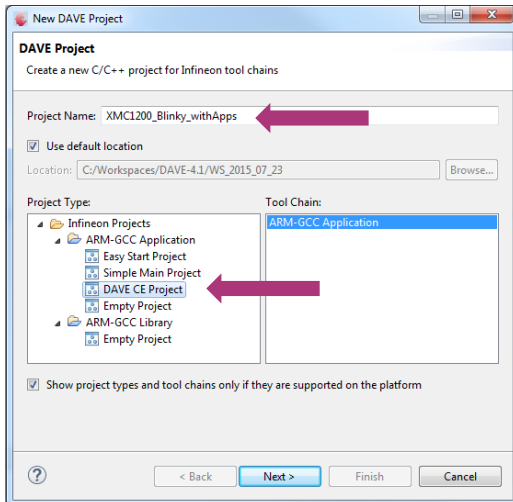
Getting Started – Example – Blinky based on DAVE™ APPs (1/7)

1. Open DAVE™

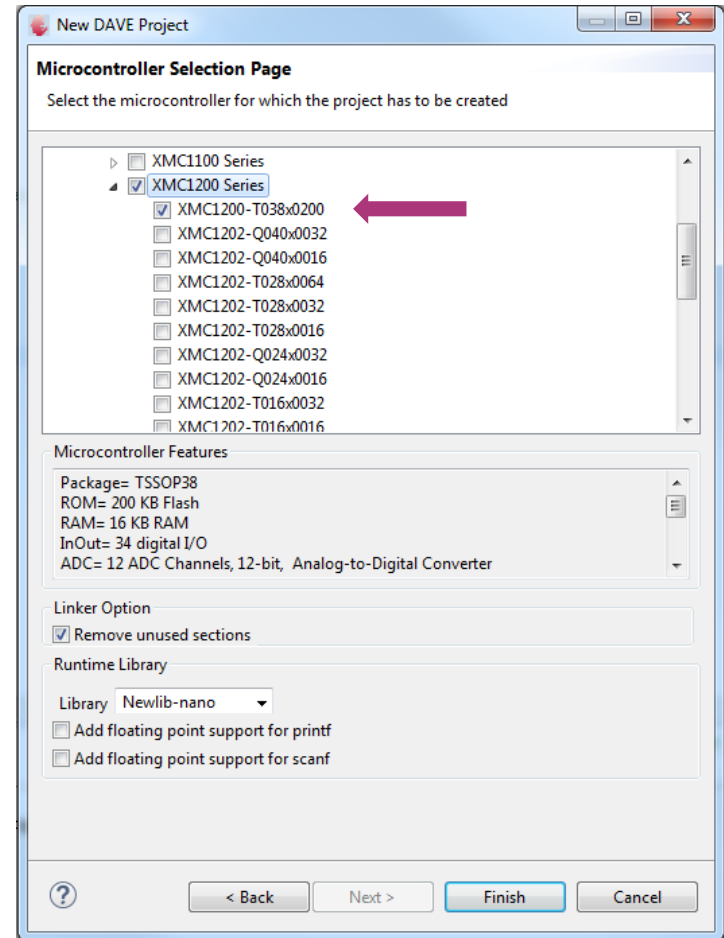


2. Create a new "DAVE CE" project:

- File → New → DAVE Project
- Enter project name e.g. "XMC1200_Blinky_withApps"
- Select "DAVE CE Project" as Project Type



3. Select the device accordingly



Getting Started – Example – Blinky based on DAVE™ APPs (2/7)




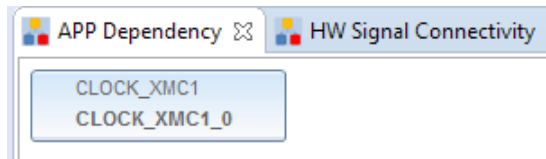
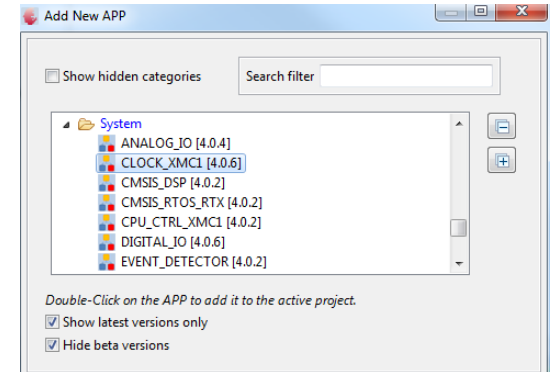
- › For this project, we will use
 - System clock frequency of 8MHz
 - LED on Port pin 0.0
 - System timer as the time base for interrupt
 - Time base of 0.2s

- › Next, we will show you how to
 1. Set up the System or Main Clock (MCLK)
 2. Configure Port pin
 3. Configure System Timer and define its exception service routine

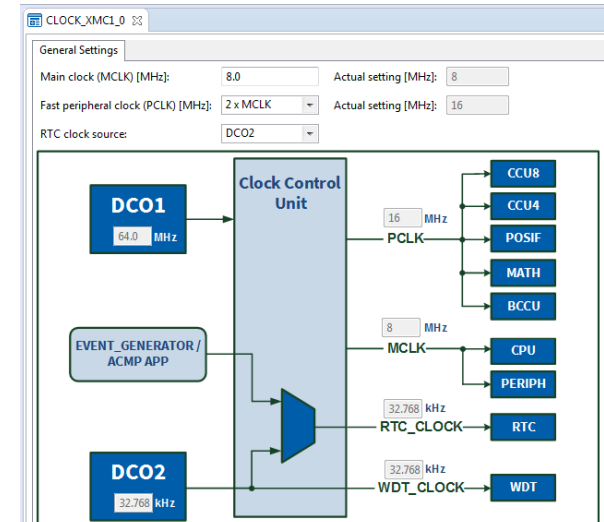
Getting Started – Example – Blinky based on DAVE™ APPs (3/7)

1. Set up System or Main Clock (MCLK)

- Click  to add new APP
- Double-click **CLOCK_XMC1** APP and close window
- Open APP configuration editor
 - In **APP Dependency** view, double-click **CLOCK_XMC1**




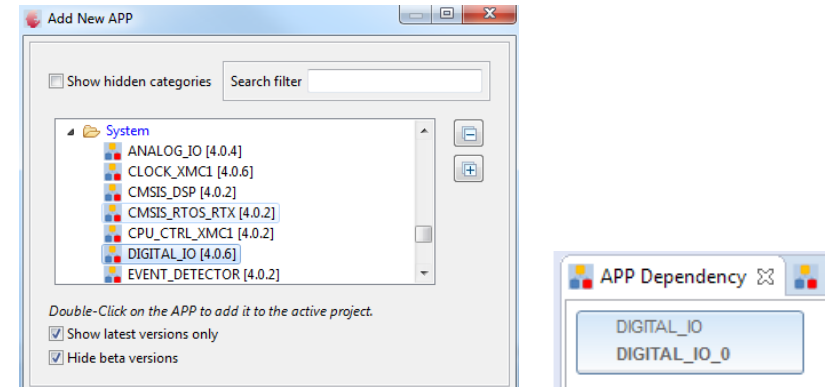
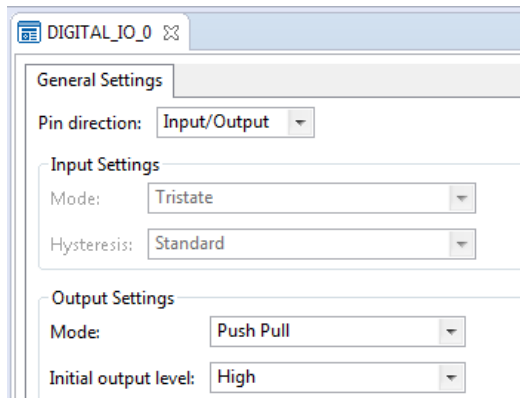
- Configure APP instance
 - In APP configuration window, set **Main clock (MCLK)** to 8MHz




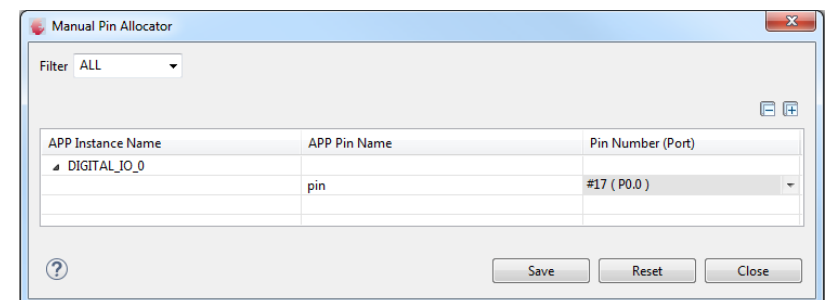
Getting Started – Example – Blinky based on DAVE™ APPs (4/7)

2. Configure Port pin


- Click  to add new APP
- Double-click **DIGITAL_IO** APP and close window
- Open APP configuration editor
 - In **APP Dependency** view, double-click DIGITAL_IO
- Configure APP instance
 - In APP configuration window, set **Pin direction** to Input/Output and set **Initial output level** to High

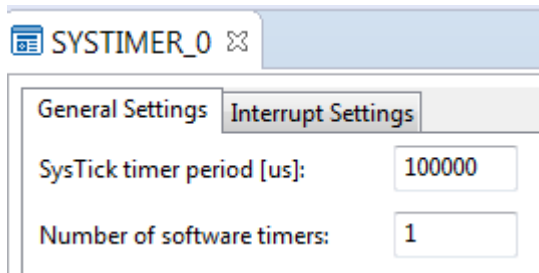
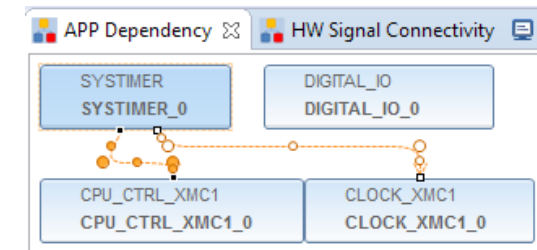
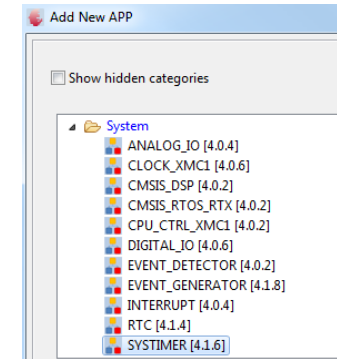


- Assign pin to P0.0
 - Click  to open **Manual Pin Allocator**
 - Set Pin Number (Port) to **#17 (P0.0)**
 - **Solve and Save**



Getting Started – Example – Blinky based on DAVE™ APPs (5/7)

3. Configure System Timer and define its exception service routine
 - Click  to add new APP
 - Double-click **SYSTIMER** APP and close window
 - Open APP configuration editor
 - In **APP Dependency** view, double-click SYSTIMER
 - Configure APP instance
 - In APP configuration window, under **General Settings** tab, set **System timer tick interval** to 100000us (0.1s)



Getting Started – Example – Blinky based on DAVE™ APPs (6/7)



- Create software timer using SYSTIMER Apps.

```
TimerId = (uint32_t)SYSTIMER_CreateTimer(Point2SEC,SYSTIMER_MODE_PERIODIC,(void*)LED_Toggle_EveryPoint2Sec,NULL);  
if (TimerId != 0U)  
{  
    //timer is created successfully, now start/run software timer  
    status = SYSTIMER_StartTimer(TimerId);  
}
```

- Define exception handler routine in **Main.c**
 - Define the toggle interval (in usec)

```
#define Point2SEC 200000U  
void LED_Toggle_EveryPoint2Sec(void)  
{  
    DIGITAL_IO_ToggleOutput(&DIGITAL_IO_0); //toggles : 1 -> 0 (if initial output level is logic 1)  
}
```

Getting Started – Example – Blinky based on DAVE™ APPs (7/7)

› Generate code

1. Click

› Build project

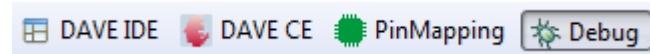
1. Click

```
'Invoking: ARM-GCC Print Size'  
"C:\DAVEv4\DAVE-4.1.2\eclipse\ARM-GCC-49\bin/arm-none-eabi-size" -  
text    data    bss     dec     hex filename  
3936    20      1096    5052    13bc XMC1200_Blinky_withApps.elf  
'Finished building: XMC1200_Blinky_withApps.siz'
```

2. Wait for Build to finish

› Download code

1. Click
2. Switch to Debug perspective
3. Click to run code



› LED blinks every 0.2s



General Information (1/2)

- › Where to buy kit:
 - <http://ehitex.com/starter-kits/for-xmc1000>
 - Order Number: KIT_XMC12_BOOT_001

- › Infineon parts utilized on kit:

Infineon Parts	Order Number
XMC1200 Microcontroller	XMC1200-T038F0200
XMC4200 Microcontroller	XMC4200-Q48F256
3V3 regulator	IFX25001MEV33

General Information (2/2)

- › Kit documentation:
 - [Boot Kit XMC1200](#)

- Video Series: XMC1000 Boot Kit Getting Started
 - [Introduction](#)
 - [DAVE™ Setup](#)
 - [Boot Mode Index Configuration via DAVE or MemTool](#)
 - [XMC1200 Hardware Setup](#)
 - [Simple Blinky Example](#)
 - [Blinky Example based on DAVE™ Apps](#)
 - [Example Projects Download](#)

References – Where to find XMC Lib documentation?

1. Go to DAVE™ Version 4 website

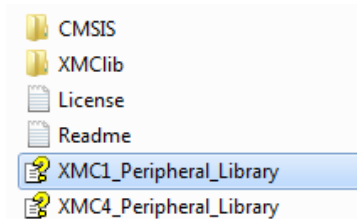
<http://www.infineon.com/dave/v4>

2. Download XMC Lib and unzip file

XMC™ Lib Ready to use APIs for peripherals which are tested for GNU-, ARM-, IAR- und TASKING- compiler, and released for Altium, ARM/KEIL, Atollic, IAR Systems und Rowley compiler IDEs. Low level driver libraries for XMC™ peripherals (APIs), CMSIS / MISRA 2004 compliant including documentation. XMC™ Lib – [Release Note](#)

System:	Timer/PWM:	Analog-mixed Signal:	Communication:	Application specific:	Examples:
<ul style="list-style-type: none"> ■ DMA ■ ERU ■ FCE ■ FLASH ■ GPIO ■ MATH ■ PAU ■ PRNG 	<ul style="list-style-type: none"> ■ CCU4 ■ CCU8 ■ HRPWM ■ POSIF 	<ul style="list-style-type: none"> ■ ACMP ■ ADC ■ DAC ■ DSD 	<ul style="list-style-type: none"> ■ CAN ■ I2C ■ SPI ■ UART ■ USB ■ USIC ■ Ethernet 	<ul style="list-style-type: none"> ■ BCCU ■ LEDTS ■ MATH ■ POSIF ■ HRPWM 	<ul style="list-style-type: none"> ■ Examples for all peripherals drivers and ARM, GCC, IAR, and Tasking

3. Open XMC1_Peripheral_Library



4. Click on **Modules** or **Files**

XMC Peripheral Library for XMC1000 Family
2.0.0

Main Page Related Pages **Modules** Files

XMC Peripheral Library

Table of Contents

- ↓ Supported devices and toolchains
- ↓ Overview
- ↓ Coding Rules and Conventions
- ↓ How to use the XMC Peripheral Library
 - ↓ Device Names
 - ↓ Directories and Files
 - ↓ XMC Lib examples
 - ↓ Keil MDK-ARM
 - ↓ IAR Embedded Workbench for ARM
 - ↓ DAVE
 - ↓ MISRA-C 2004 Compliance Exceptions
 - ↓ Test conditions
 - ↓ XMC Peripheral Library Licensing

The XMC Peripheral Library (XMC Lib) consists of low-level drivers for the XMC product family peripherals. Built on top of the Cortex Microcontroller Software Interface Standard (CMSIS) and MISRA-C 2004 compliant, it provides access to all peripheral features.

Main Page **Modules** Files

Modules

Here is a list of all modules:

- ▶ XMC Peripheral Library
 - ACMP Analog Comparator(ACMP) low level driver for XMC family of microcontrollers.
 - BCCU Brightness and Color Control Unit (BCCU) driver for the XMC1 microcontroller family
 - CCU4 Capture Compare Unit 4 (CCU4) low level driver for XMC family of microcontrollers
 - CCU8 Capture Compare Unit 8 (CCU8) low level driver for XMC family of microcontrollers
 - ERU Event Request Unit (ERU) driver for the XMC microcontroller family
 - FLASH Flash driver for XMC microcontroller family
 - GPIO General Purpose Input Output (GPIO) driver for the XMC microcontroller family
 - I2C Inter Integrated Circuit(I2C) driver for the XMC microcontroller family
 - LEDTS LED and Touch-Sense control(LEDTS) driver for the XMC controller family
 - MATH MATH Coprocessor (MATH) driver for the XMC1302 microcontroller family
 - PAU Peripheral Access Unit (PAU) driver for the XMC1000 microcontroller family
 - POSIF Position Interface Unit (POSIF) driver for the XMC microcontroller family
 - PRNG Pseudo Random Number Generator (PRNG) driver for XMC1000 microcontroller family
 - RTC RTC driver for XMC microcontroller family
 - SCU System Control Unit(SCU) driver for XMC microcontroller family
 - SPI Synchronous serial channel driver for SPI-like communication
 - UART Universal Asynchronous Receiver/Transmitter (UART) driver for XMC microcontroller family
 - USIC Universal Serial Interface Channel(USIC) driver for serial communication
 - VADC Versatile Analog to Digital Converter (VADC) driver for XMC microcontroller family
 - WDT Watchdog driver for the XMC microcontroller family

Main Page **Modules** **Files**

File List

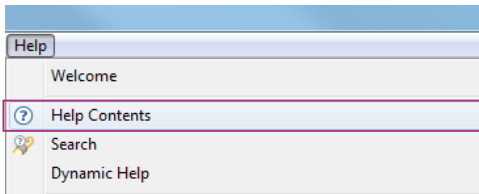
Here is a list of all documented files with brief descriptions:

- ↓ xmc1_flash.h
- ↓ xmc1_gpio.h
- ↓ xmc1_rtc.h
- ↓ xmc1_scu.h
- ↓ xmc_acmp.h
- ↓ xmc_bccu.h
- ↓ xmc_ccu4.h
- ↓ xmc_ccu8.h
- ↓ xmc_eri.h
- ↓ xmc_flash.h
- ↓ xmc_gpio.h
- ↓ xmc_i2c.h
- ↓ xmc_ledts.h
- ↓ xmc_math.h
- ↓ xmc_pau.h
- ↓ xmc_posif.h

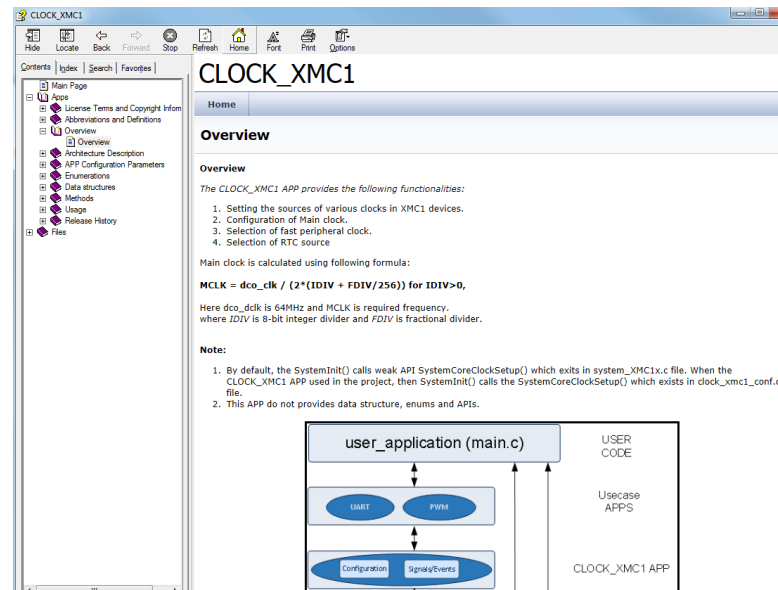
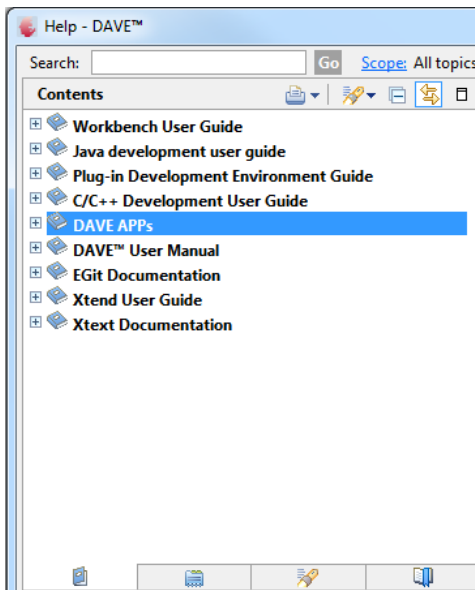


References – Where to find DAVE™ APP documentation?

1. In DAVE™, go to Help → Help Contents



2. Expand DAVE Apps → Click on **CLOCK_XMC1** → **Overview**



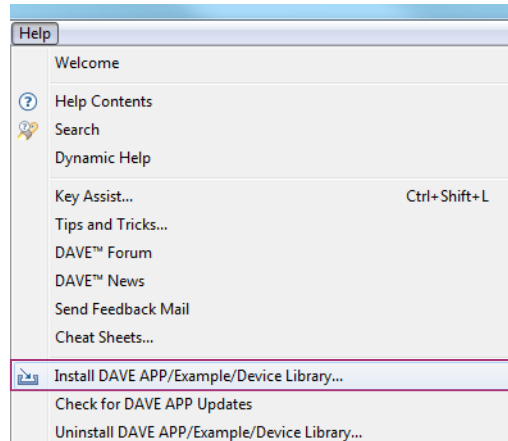
References – Where to download DAVE™ Example Projects?

1. Example Project library within DAVE™
2. DAVE™ website
3. Example from XMC Lib package

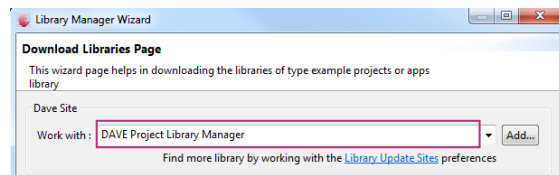
References – How to load Example Project in DAVE™? (1/4)

› Example Project library within DAVE™

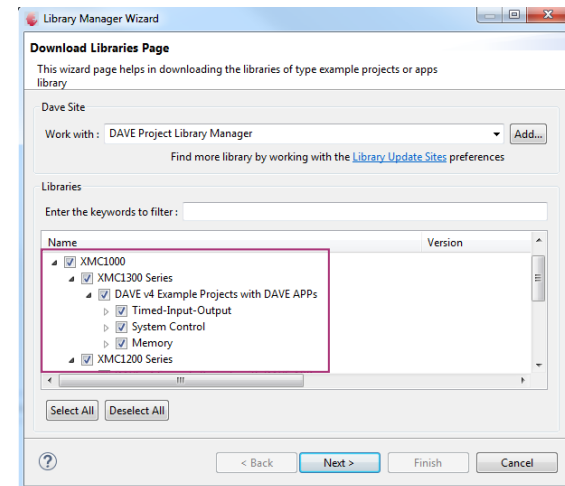
1. Help → Install DAVE APP/Example/Device Library



2. Select DAVE Project Library Manager



3. Select Examples in the Libraries window → Click Next



4. Accept terms of the license agreements → Click Finish

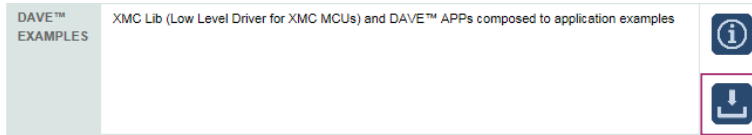
References – How to load Example Project in DAVE™? (2/4)

› DAVE™ website

1. Go to DAVE™ Version 4 website

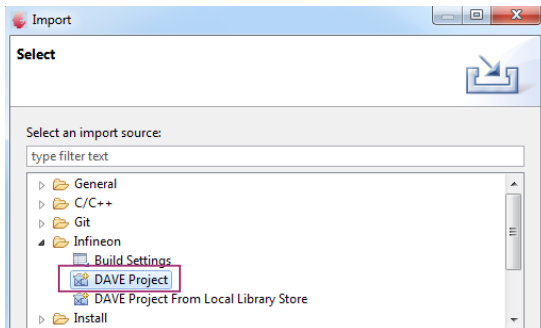
<http://www.infineon.com/dave/v4>

2. Download DAVE™ EXAMPLES

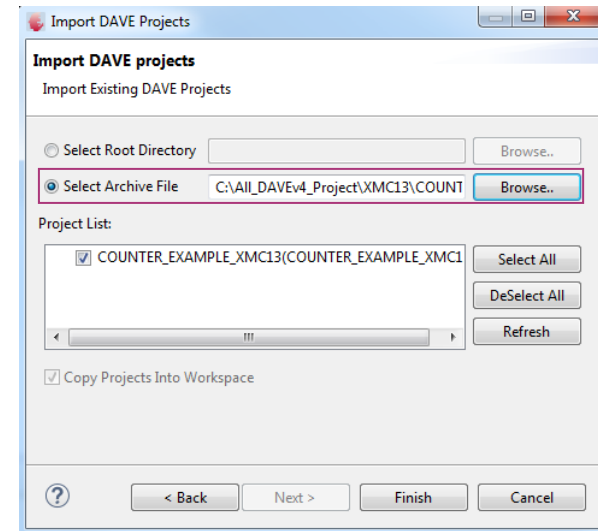


3. In DAVE™, go to File → Import

4. Select DAVE Project → Next



5. Select Archive File → Browse to downloaded project zip file



6. Click Finish

References – How to load Example Project in DAVE™? (3/4)


› Example from XMC Lib package

1. Go to DAVE™ Version 4 website

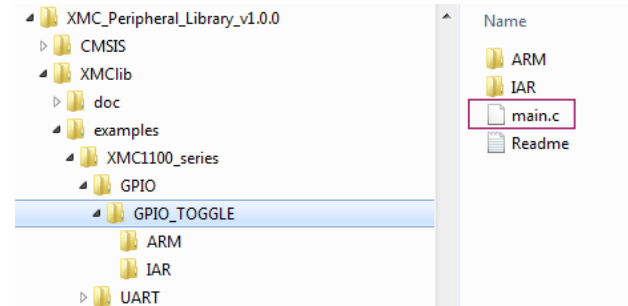
<http://www.infineon.com/dave/v4>

2. Download XMC Lib and unzip file

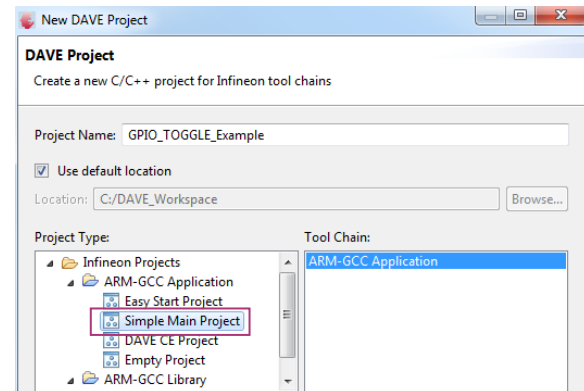
System:	Timer/PWM:	Analog-mixed Signal:	Communication:	Application specific:	Examples:
<ul style="list-style-type: none">DMAERUFCEFLASHGPIOMATHPAUPRNG	<ul style="list-style-type: none">CCU4CCU8HRPWMPOSIF	<ul style="list-style-type: none">ACMPADCDAC	<ul style="list-style-type: none">CANI2CSPIUARTUSBUSIC	<ul style="list-style-type: none">BCCULEDTSMATHPOSIFHRPWM	<ul style="list-style-type: none">Examples for all peripherals drivers and ARM, GCC, IAR, and Tasking



3. Example code (main.c) can be found within XMC Lib package

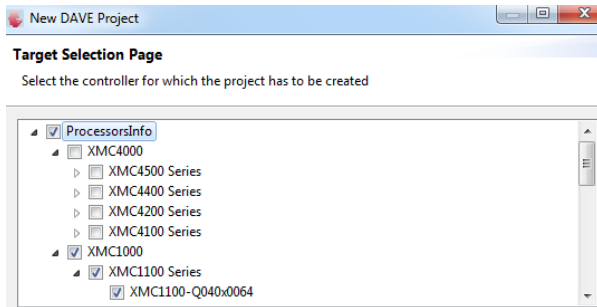


4. Create new “Simple Main Project” in DAVE™



References – How to load Example Project in DAVE™? (4/4)

5. Select target device of selected main.c example

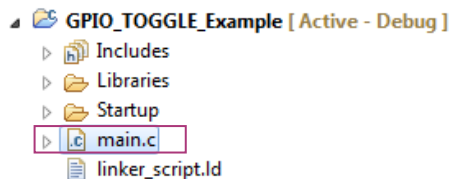


7. Click  to Build project

8. Click  to download and run project on target board

6. Delete main.c in the newly created DAVE project

7. Copy main.c from XMC Lib example into DAVE project





Part of your life. Part of tomorrow.

