Getting Started - XMC1400 Boot Kit

XMC Microcontrollers

Dec 2015

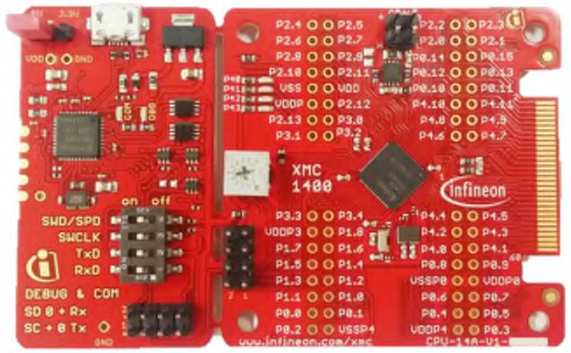
Agenda

1. Kit Overview
2. Hardware Overview
3. Tooling Overview
4. Getting Started
5. Resource Listing
6. References

Kit Overview (1/2)

- XMC1400 Boot Kit
  - Consists of an XMC1400 CPU Card
  - Supported Application Cards examples: Colour LED Card, White LED Card

(Application Card is orderable separately or as part of another Application Kit)



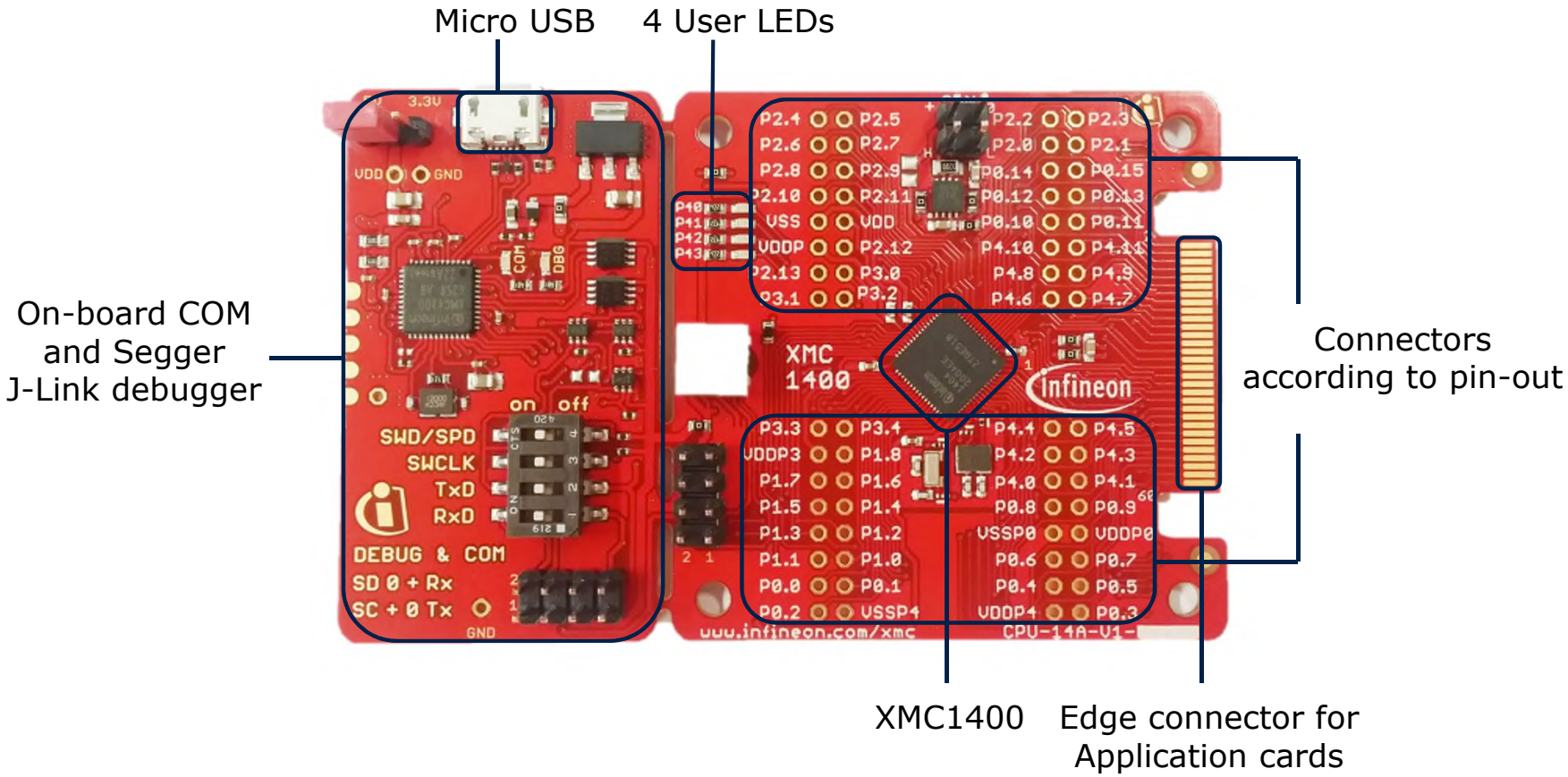
XMC1400 CPU Card

Colour LED Card

White LED Card

Kit Overview (2/2)

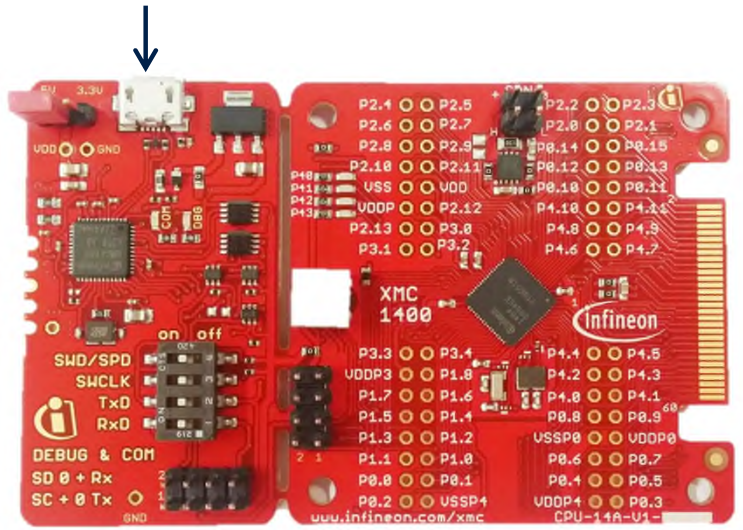
› XMC1400 CPU Card



Hardware Overview (1/2)

- Connect XMC1400 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

Home

Hardware Overview (2/2)

- Kit information

Nr.	Kit Name	Kit Description	Order Number
1	KIT_XMC14_BOOT_001	Boot Kit XMC1400	KIT_XMC14_BOOT_001

- Infineon parts utilized on Kit Nr. 1:

Infineon Parts	Order Number
XMC1400 Microcontroller	XMC1404-Q064X0200
XMC4200 Microcontroller	XMC4200-Q48F256
3V3 regulator	IFX25001MEV33

Tooling Overview - Boot Modes

- Boot Modes available
  - CAN Bootstrap-Loader Mode
  - 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™ v4.1.4 download
  - MemTool
    - Download MemTool

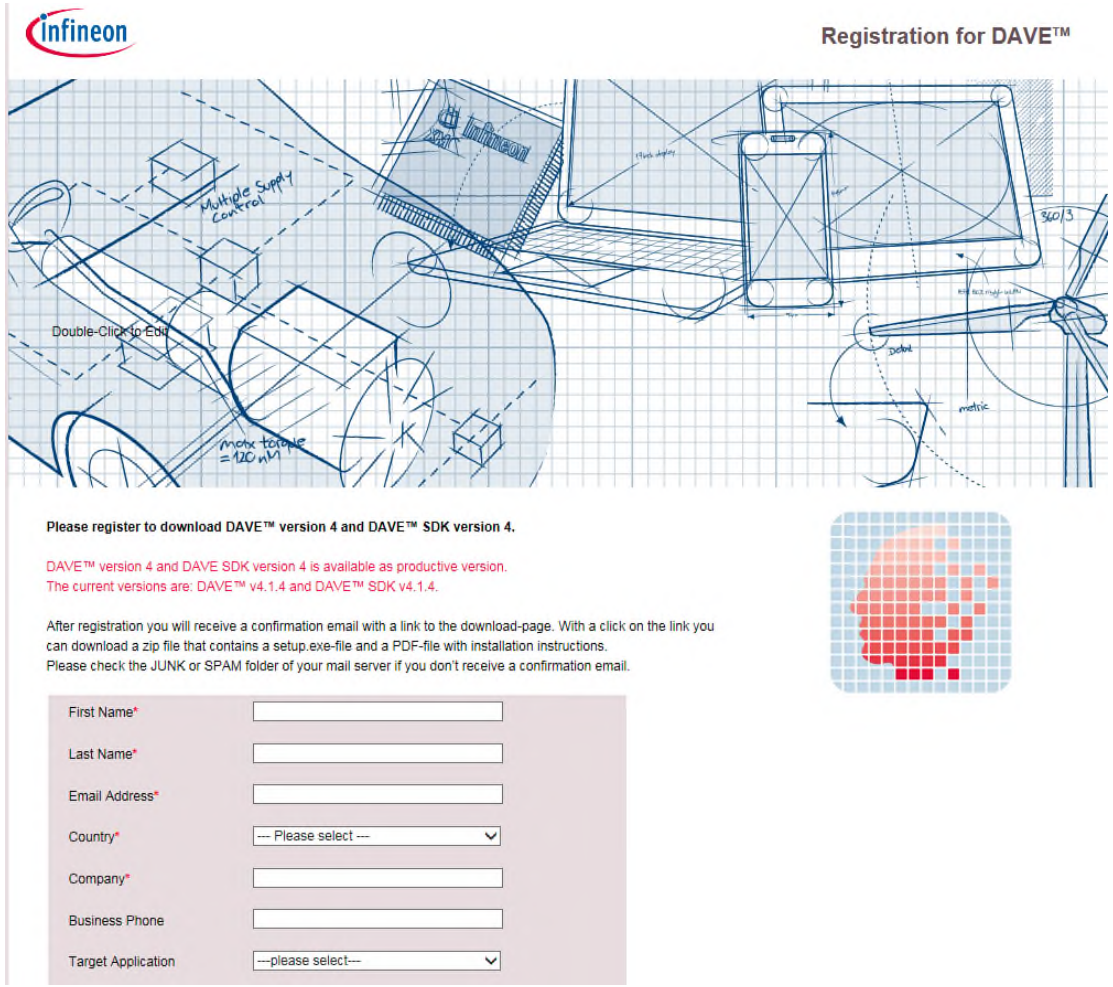
    MemTool v4.65.exe download

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

Home

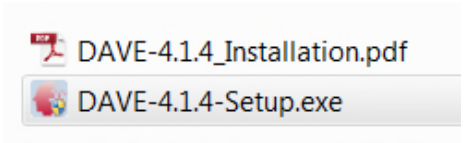
Tooling Overview – DAVE™ (1/5)

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

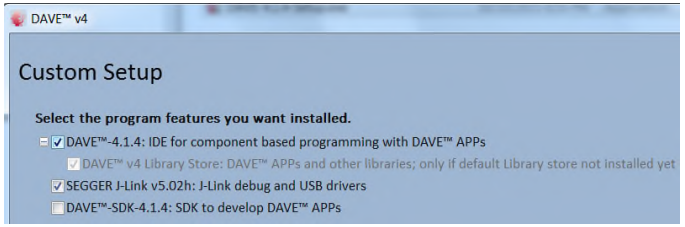


Tooling Overview – DAVE™ (2/5)

- After registration, download and unzip the installer package

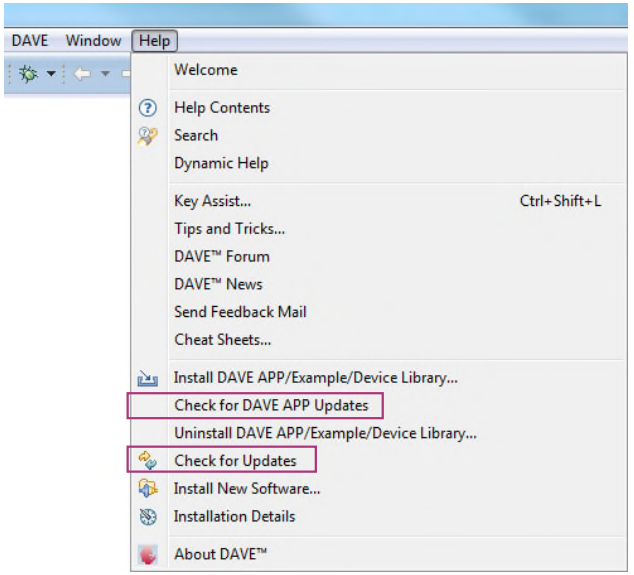


- Run DAVE-4.1.4-Setup.exe to install DAVE™ IDE and SEGGER J-Link drivers



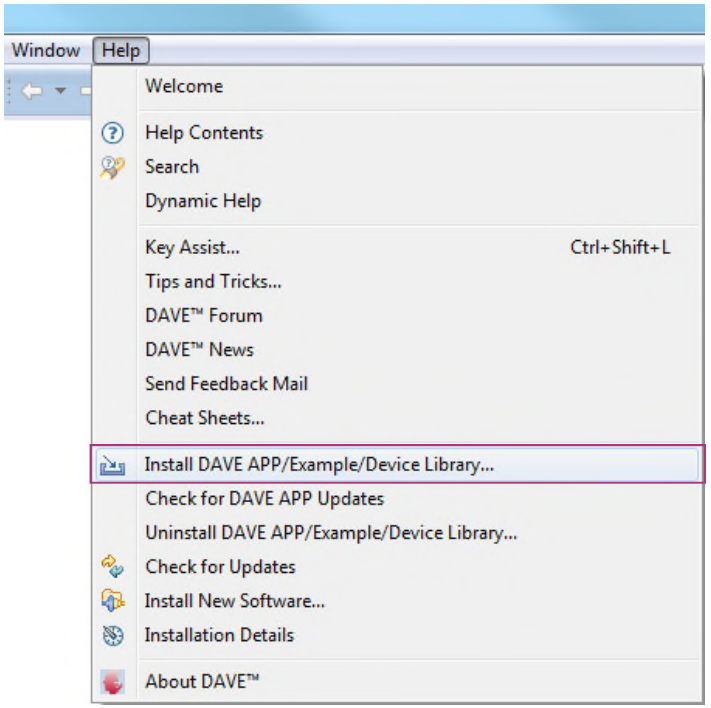
- Open DAVE™

- Update DAVE™ and DAVE™ libraries
  - Help → Check for Updates
  - Help → Check for DAVE APP Updates



Tooling Overview – DAVE™ (3/5)

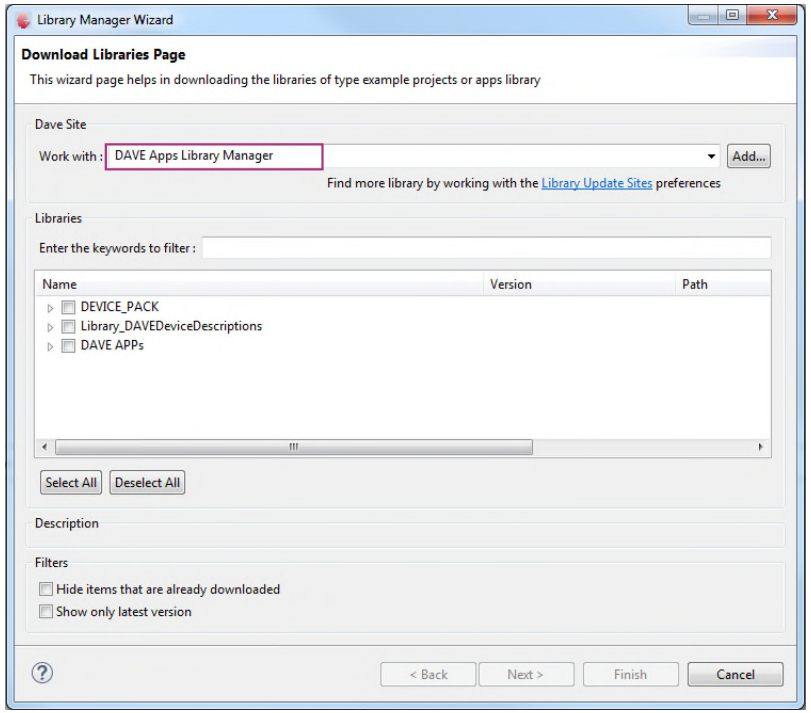
- Install DAVE™ APPs libraries and Device Description
  - Help → Install DAVE APP/Example/Device Library



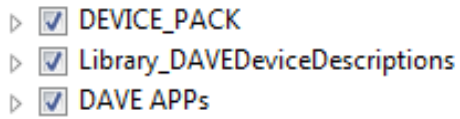
- Note: You may skip the above step if you are not using DAVE™ APPs

Tooling Overview – DAVE™ (4/5)

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

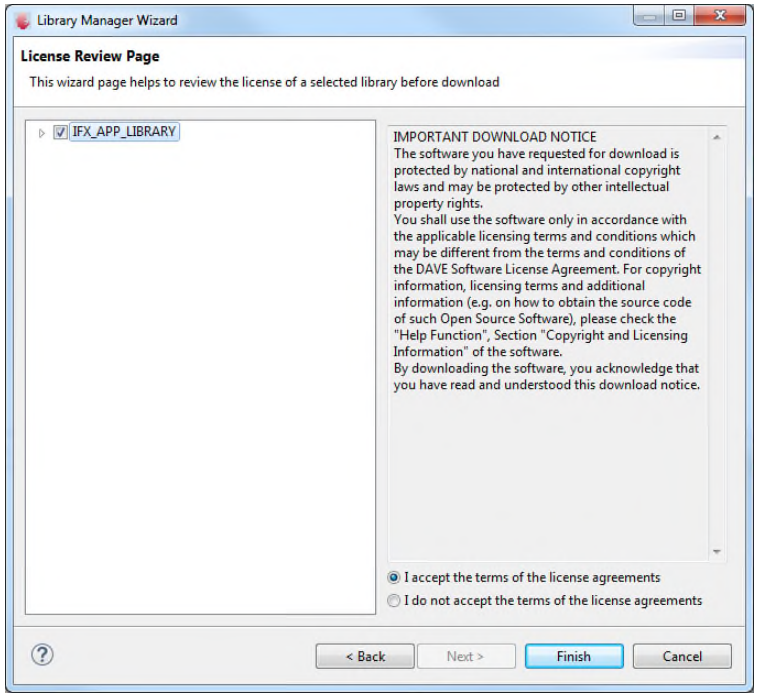


- Select DEVICE_PACK, Library_DAVEDeviceDescriptions (XMC1400 Device) and DAVE APPs



Tooling Overview – DAVE™ (5/5)

- Accept terms of the license agreements and click Finish



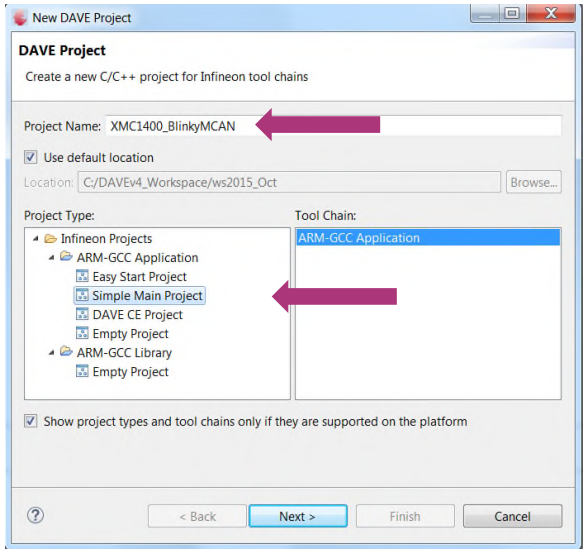
- DAVE™ APPs libraries and Device Description are installed

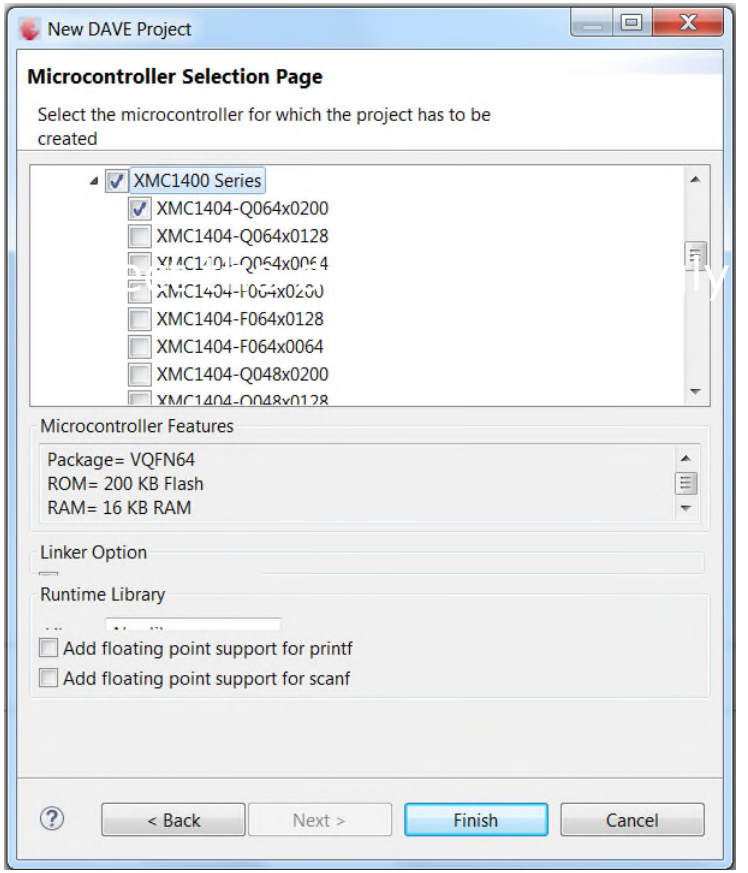
Home

Getting Started – Example – Blinky based on XMC Lib (1/8)

1. Open DAVE™ 

2. Create a new "Simple Main" project:
   - File → New → DAVE Project
   - Enter project name e.g. "XMC1400_BlinkyMCAN"
   - Select "Simple Main Project" as Project Type



3. 

Getting Started – Example – Blinky based on XMC Lib (2/8)

- For this project, we will use
  - System clock frequency of 8MHz
  - LED on Port pin
  - System timer, SysTick, as the time base for interrupt
  - P4.9 for CAN Tx and P4.8 for CAN Rx

- 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
  4. Configure 2 CAN message objects
  5. Configure P4.9 for CAN Tx and P4.8 for CAN Rx

Getting Started – Example – Blinky based on XMC Lib (3/8)

1. Set up System or Main Clock (MCLK) and include the required header files.

```
#include "xmc_can.h"
#include "xmc_gpio.h"
```

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

```
XMC_SCU_CLOCK_CONFIG_t clock_config =
{
    .dclk_src = XMC_SCU_CLOCK_DCLKSRC_DCO1,
    .oschp_mode = XMC_SCU_CLOCK_OSCHP_MODE_OSC,
    .enable_automatic_dco1_calibration = true,
    .clock_sync = XMC_SCU_CLOCK_SYNC_CLKSRC_OSCHP, //XMC_SCU_CLOCK_SYNC_CLK_OSCLP,
    .pclk_src = XMC_SCU_CLOCK_PCLKSRC_DOUBLE_MCLK,
    .fdiv = 0,  //Fractional divider
    .idiv = 1   //MCLK = 48MHz
};
```

   - Initializes clock generators ad clock tree in Main.c to set MCLK = 48MHz and PCLK = 96MHz

```
XMC_SCU_CLOCK_Init(&clock_config);
```

Getting Started – Example – Blinky based on XMC Lib (4/8)

2. Configure Port pin

- GPIO to toggle the LED is configured via mode and output_level of XMC_GPIOC_CONFIG structure. P4.0 will toggle at regular interval. P4.1 will toggle when CAN message transmit and P4.3 toggle when CAN message received.

```
#define LED P4_0
#define LED_TX P4_1  //CAN TX indication
#define LED_RX P4_3  //CAN_RX indication

led_config.mode = XMC_GPIO_MODE_OUTPUT_PUSH_PULL;
XMC_GPIO_Init(LED, &led_config);
led_TX_config.mode = XMC_GPIO_MODE_OUTPUT_PUSH_PULL;
led_TX_config.output_level = XMC_GPIO_OUTPUT_LEVEL_HIGH;
XMC_GPIO_Init(LED_TX, &led_TX_config);
led_RX_config.mode = XMC_GPIO_MODE_OUTPUT_PUSH_PULL;
led_RX_config.output_level = XMC_GPIO_OUTPUT_LEVEL_HIGH;
XMC_GPIO_Init(LED_RX, &led_RX_config);
```

- Configure P4.9 as CAN Tx pin and P4.8 as CAN Rx pin.

```
XMC_GPIO_CONFIG_t CAN1_TXD_config;
XMC_GPIO_CONFIG_t CAN1_RXD_config;
CAN1_TXD_config.mode = XMC_GPIO_MODE_OUTPUT_PUSH_PULL_ALT9;
CAN1_RXD_config.mode = XMC_GPIO_MODE_INPUT_TRISTATE;
XMC_GPIO_Init(CAN1_TXD, &CAN1_TXD_config);
XMC_GPIO_Init(CAN1_RXD, &CAN1_RXD_config);
```

```
#define CAN1_TXD P4_9 //ALT9
#define CAN1_RXD P4_8
```

Getting Started – Example – Blinky based on XMC Lib (5/8)

3. Configure SysTick and define its exception service routine

   - SysTick exception handler is defined in startup_XMC1300.s by using the macro.

```
.macro Insert_InterruptVeener Interrupt
    .globl \Interrupt\()_Veener
\Interrupt\()_Veener:
    LDR R0, =\Interrupt\()_Handler
    BX  R0
.endm
```

   - Initialize the SysTick in Main.c

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

```
#define TICKS_PER_SECOND 1000
#define TICKS_WAIT 500
```

   - Define the SysTick exception handler routine. Every 0.5s, toggle P4.0 and transmit the CAN message, followed by toggle P4.1.

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

  ticks++;
  if (ticks == TICKS_WAIT)
  {
    XMC_GPIO_ToggleOutput(LED);
    XMC_CAN_MO_Transmit(&MCAN_message1);
    XMC_GPIO_ToggleOutput(LED_TX);
    ticks = 0;
  }
}
```

Getting Started – Example – Blinky based on XMC Lib (6/8)

4. Configure CAN node 1 baudrate and CAN message object

```
/*CAN Bit time*/
XMC_CAN_NODE_NOMINAL_BIT_TIME_CONFIG_t baud = {
  .can_frequency=11000000,
  .baudrate=500000,
  .sample_point =6000,
  .sjw=3,
};
```

- Configure Tx message object and its data bytes

```
/*CAN message= CAN_MO4 */
XMC_CAN_MO_t MCAN_message1 = {
  .can_mo_ptr = CAN_MO4,
  .can_priority = XMC_CAN_ARBITRATION_MODE_IDE_DIR_BASED_PRIO_2,
  .can_identifier = 0x3ff,
  .can_id_mask= 0x7ff,
  .can_id_mode = XMC_CAN_FRAME_TYPE_STANDARD_11BITS,
  .can_ide_mask = 1,
  .can_data_length = 8,
  .can_data = {0x12345555, 0x12345556},
  .can_mo_type= XMC_CAN_MO_TYPE_TRANSMSGOBJ
};
```

- Configure Rx message object

```
/*CAN message= CAN_MO2 */
XMC_CAN_MO_t MCAN_message2 =
{
    .can_mo_ptr = CAN_MO2,
    .can_priority = XMC_CAN_ARBITRATION_MODE_IDE_DIR_BASED_PRIO_2,
    .can_identifier = 0x2ff,
    .can_id_mask = 0x2ff,
    .can_id_mode = XMC_CAN_FRAME_TYPE_STANDARD_11BITS,
    .can_ide_mask = 1,
    .can_data_length = 8,
    .can_mo_type = XMC_CAN_MO_TYPE_RECMSGOBJ
};
```

Getting Started – Example – Blinky based on XMC Lib (7/8)

5. Set CAN Rx interrupt service request number and allocate the message objects to CAN node 1.

```
/* Set receive interrupt Service request number*/
XMC_SCU_SetInterruptControl(7, XMC_SCU_IRQCTRL_7_CAN0_SR3);
XMC_CAN_MO_SetEventNodePointer(&MCAN_message2,XMC_CAN_MO_POINTER_EVENT_RECEIVE,3);
NVIC_SetPriority(IRQ7_IRQn,1);
NVIC_EnableIRQ(IRQ7_IRQn);

/* Allocate MO in Node List*/
XMC_CAN_AllocateMOtoNodeList(CAN,1,4);
XMC_CAN_AllocateMOtoNodeList(CAN,1,2);
XMC_CAN_NODE_ResetInitBit(CAN_NODE1);
```

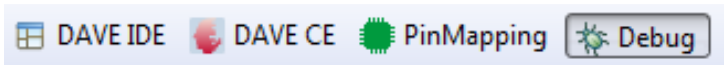
– Define the interrupt event handler for CAN node

```
  /*This function is the Interrupt Event Handler for the CAN Node*/
void  IRQ7_Handler(void)
{
  /* Toggle LED Pin 4.3  to indicate that the requested message is received*/
  XMC_CAN_MO_Receive(&MCAN_message2);
  XMC_GPIO_ToggleOutput(LED_RX);
  NVIC_ClearPendingIRQ(IRQ7_IRQn);
}
```

Getting Started – Example – Blinky based on XMC Lib (8/8)

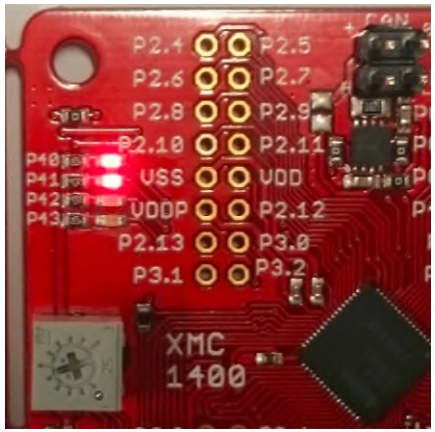
- Build project
  1. Click 
  2. Wait for Build to finish

```
'Invoking: ARM-GCC Print Size'
"C:\DAVEv4\DAVE-4.1.4\eclipse\ARM-GCC-49/bin/arm-none-eabi-size"  --format=berkeley "XMC1400_BlinkyMCAN.elf"
   text    data     bss     dec     hex filename
   3316      96    1068    4480    1180 XMC1400_BlinkyMCAN.elf
'Finished building: XMC1400_BlinkyMCAN.siz'
```

- Download code
  1. Click 
  2. Switch to Debug perspective 

  3. Click  to run code

- LED LEDs P40, P41 blinks every 0.5s

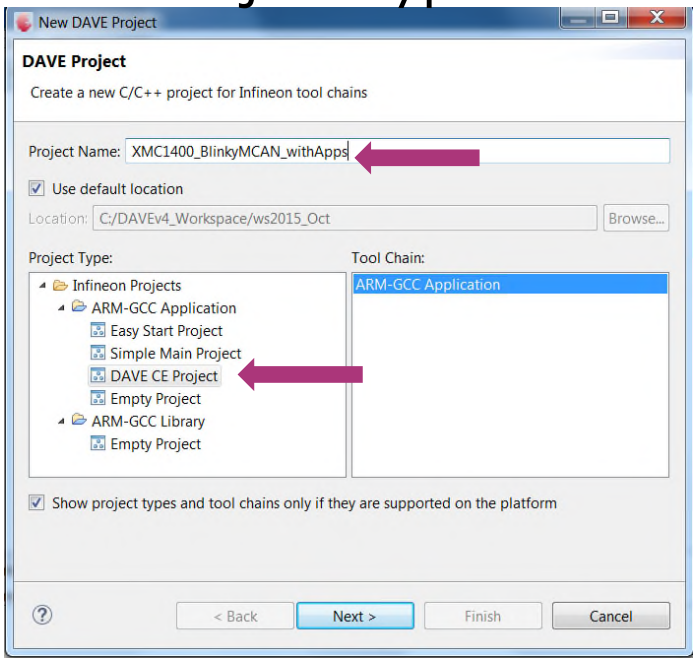


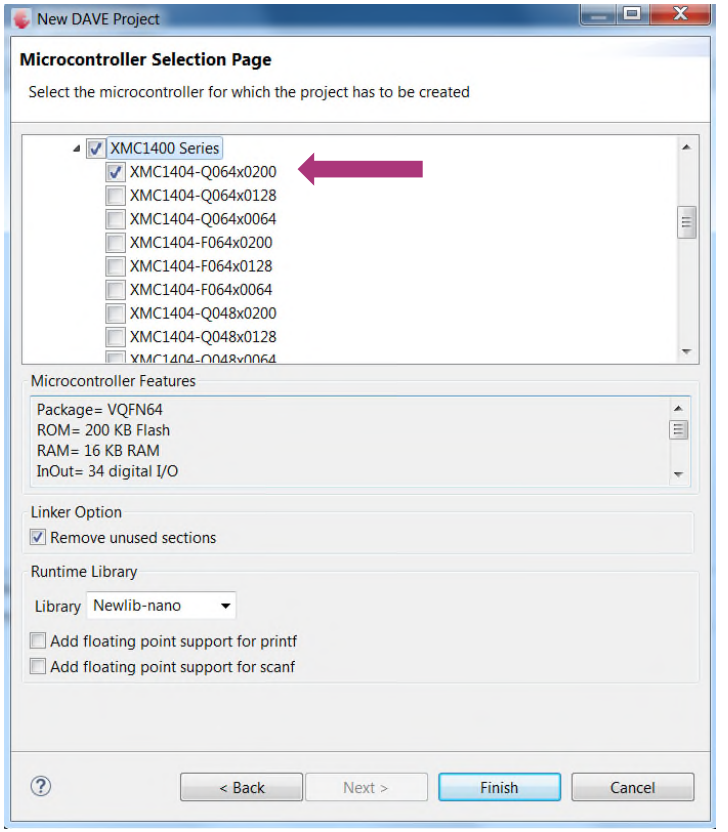
Home

Getting Started – Example – Blinky based on DAVE™ APPs (1/13)

1. Open DAVE™ 

2. Create a new "DAVE CE" project:
   - File → New → DAVE Project
   - Enter project name e.g. "XMC1400_BlinkyMCAN_withApps"
   - Select "DAVE CE Project" as Project Type



3. 

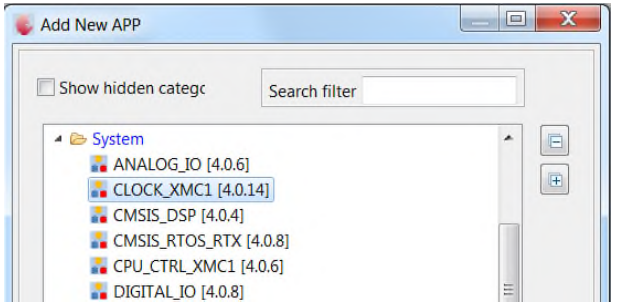
Getting Started – Example – Blinky based on DAVE™ APPs (2/13)

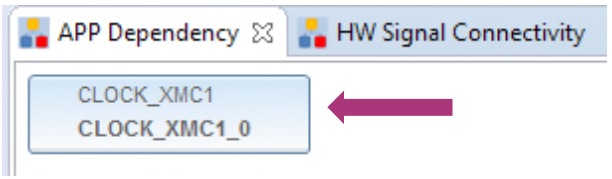
- For this project, we will use
  - System clock frequency of 48MHz
  - LED on Port pin
  - System timer as the time base for interrupt

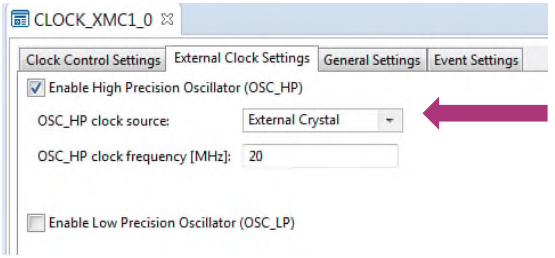
- Next, we will show you how to
  1. Set up the System or Main Clock (MCLK)
  2. Configure Port pin
  3. Configure System Timer and define its exception service routine
  4. Configure CAN Node
  5. Configure CAN Message Objects, CAN Tx/Rx pin and ISR

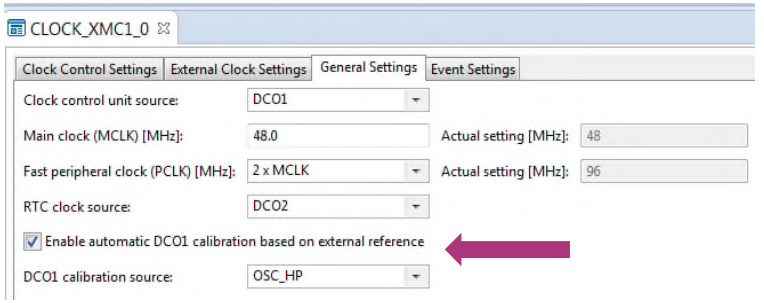
Getting Started – Example – Blinky based on DAVE™ APPs (3/13)

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

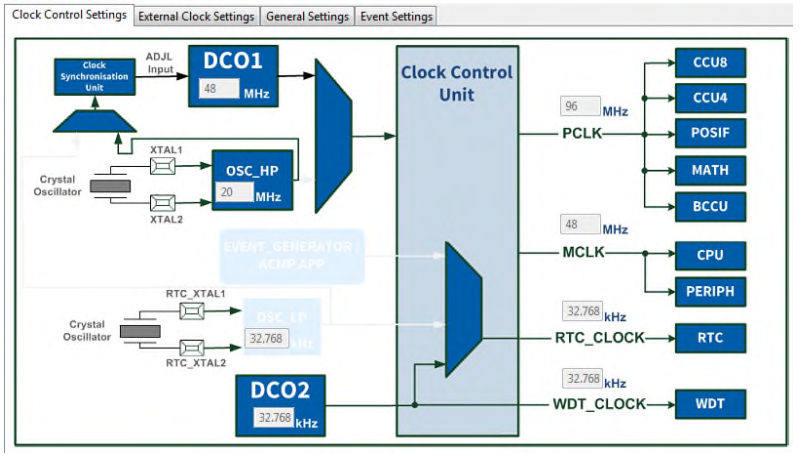








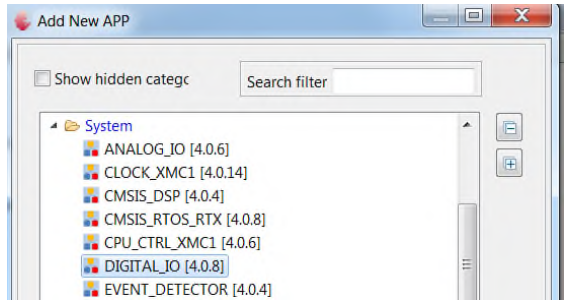
   - Configure APP instance
     - In APP configuration window, set **Main clock (MCLK)** to 48MHz

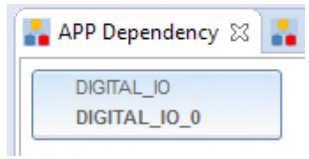


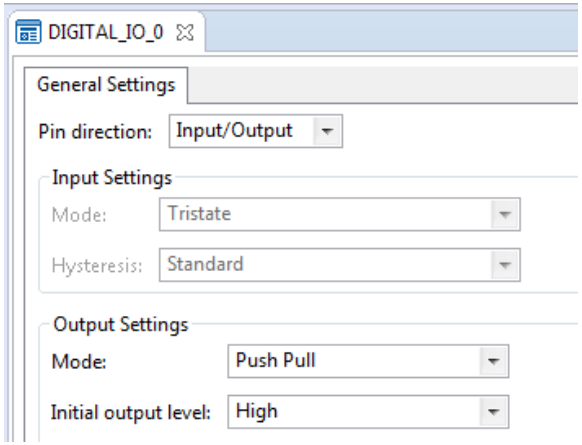
Getting Started – Example – Blinky based on DAVE™ APPs (4/13)

2. Configure Port pin

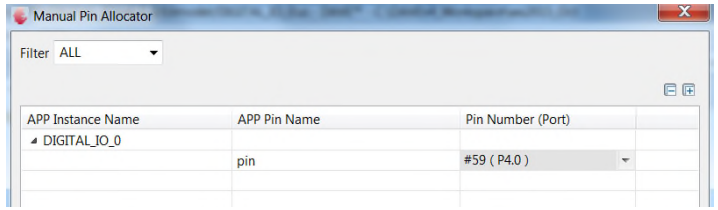
- Click  to add new APP
- Double-click **DIGITAL_IO** APP and close window
- Open APP configuration editor
  - In **APP Dependency** view, double-click DIGITAL_IO
- Configure APP instance
  - In APP configuration window, set **Pin direction** to Input/Output and set **Initial output level** to High





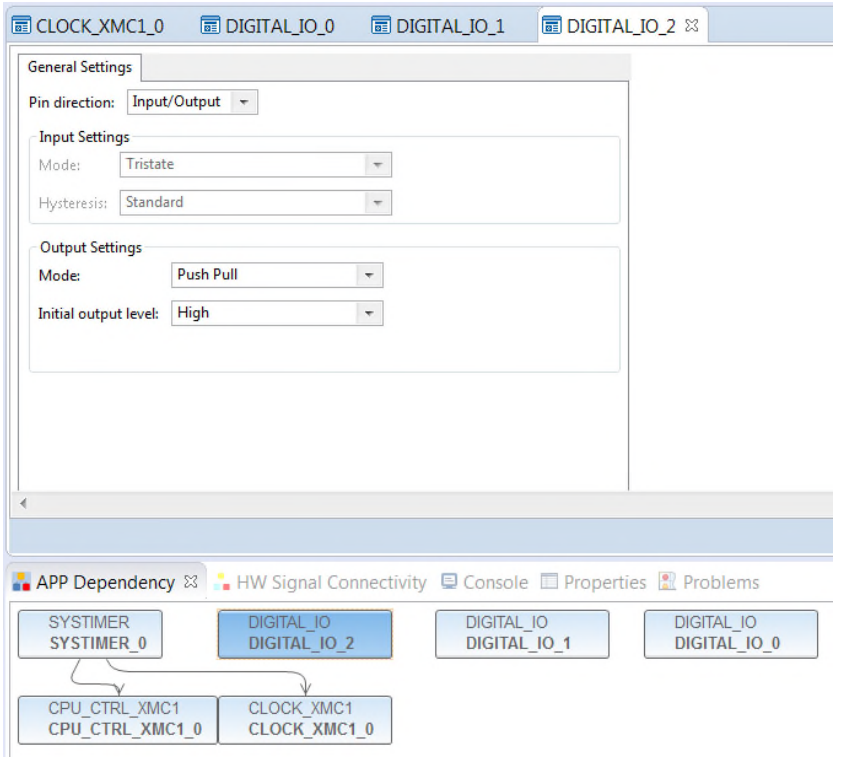


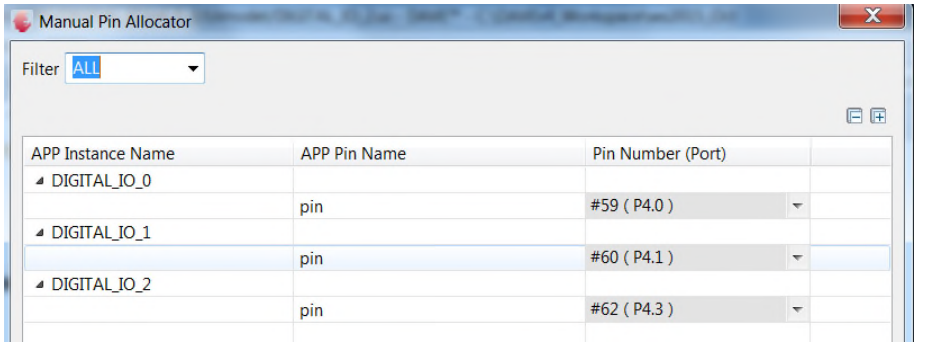
- Assign pin to P4.0
  - Click  to open **Manual Pin Allocator**
  - Set Pin Number (Port) to **#59 (P4.0)**
  - **Save and closed**
- P4.0 will toggle output at 0.5s interval



Getting Started – Example – Blinky based on DAVE™ APPs (5/13)

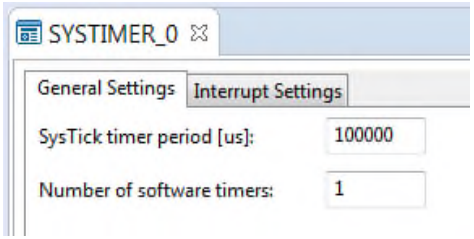
3. Configure Port pin P4.1 and P4.3 using steps described in 2.
   - P4.1 will toggle output when CAN message transmitted
   - P4.3 will toggle output when CAN message received

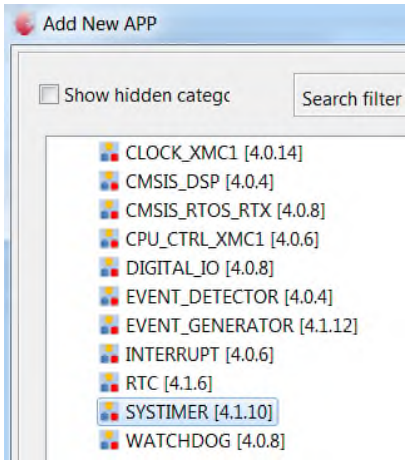


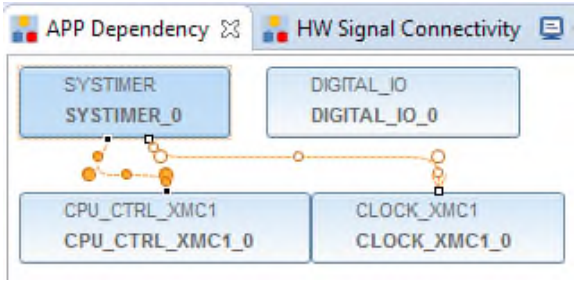


Getting Started – Example – Blinky based on DAVE™ APPs (6/13)

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 (7/13)

- Create software timer using SYSTIMER Apps.

```
uint32_t TimerId = SYSTIMER_CreateTimer(Point5SEC,SYSTIMER_MODE_PERIODIC,(void*)LED_Toggle_EveryPoint5Sec,NULL);
if(TimerId != 0U)
   {
     //Timer is created successfully
     // Start/Run Software Timer
     status = SYSTIMER_StartTimer(TimerId);
   }
```

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

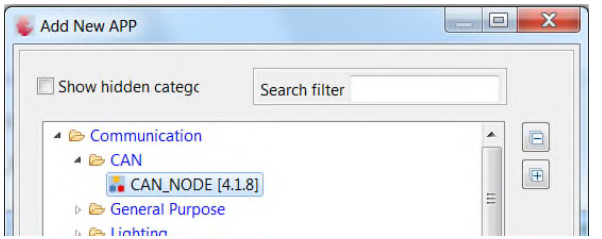
```
#define Point5SEC 500000U

void LED_Toggle_EveryPoint5Sec(void)
{
    DIGITAL_IO_ToggleOutput(&DIGITAL_IO_0);        // Toggle LED Pin 4.0 every 0.5s interval
    CAN_NODE_MO_Transmit(CAN_NODE_0.lmobj_ptr[0]); // Transmit the data of message object
    DIGITAL_IO_ToggleOutput(&DIGITAL_IO_1);        // Toggle LED Pin 4.1 to indicating transmission of data frames is done
}
```

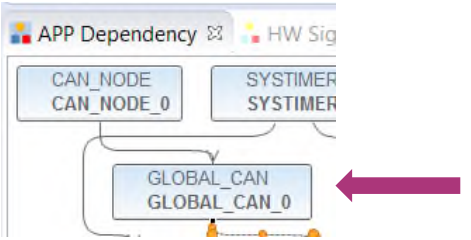
Getting Started – Example – Blinky based on DAVE™ APPs (8/13)

4. Configure CAN node

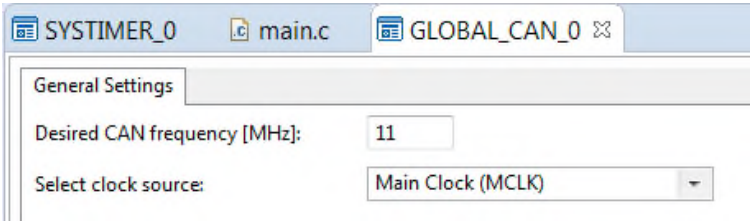
- Click  to add new APP
- Double-click **CAN_NODE** APP and close window



- Open APP configuration editor
  - In **APP Dependency** view, double-click GLOBAL_CAN



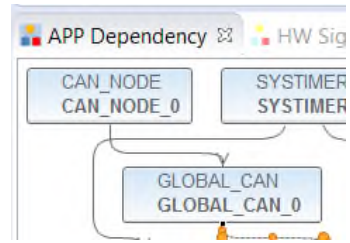
- Configure APP instance
  - In APP configuration window, under **General Settings** tab, set **Desired CAN frequency (MHz)** to 11

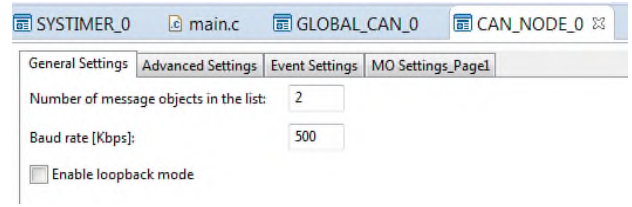


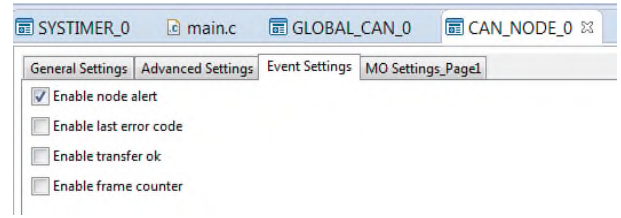
Getting Started – Example – Blinky based on DAVE™ APPs (9/13)

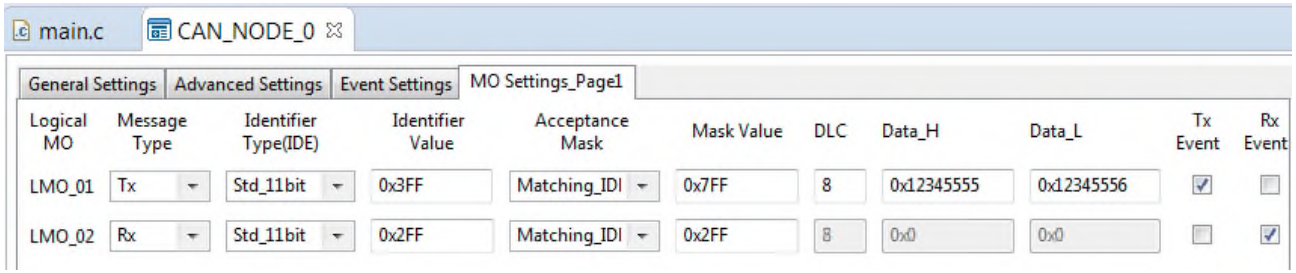
5. Configure CAN Message Objects

- Open APP configuration editor
  - In **APP Dependency** view, double-click CAN_NODE
- Configure APP instance
  - In APP configuration window, under **General Settings** tab, set **Number of message objects in the list** to 2
  - Under **Event Settings**, enable **Enable node alert**
  - Under MO Settings_Page1, setup a TX message object with 8 data bytes,Identifier value = 0x3FF
  - Under MO Settings_Page1, setup a RX message object with Identifier value = 0x2FF.



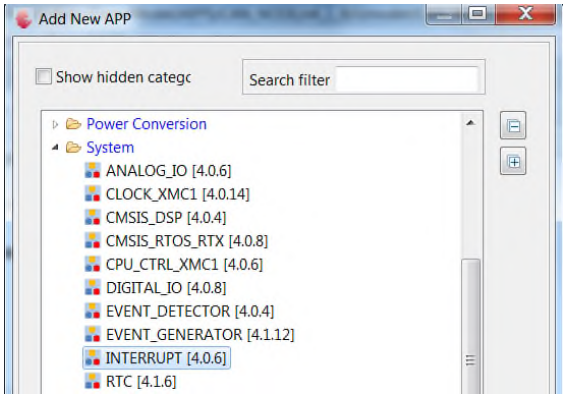




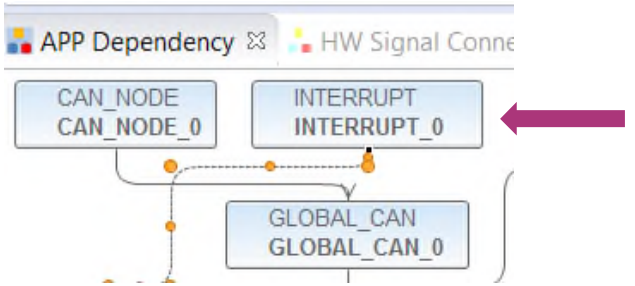


Getting Started – Example – Blinky based on DAVE™ APPs (10/13)

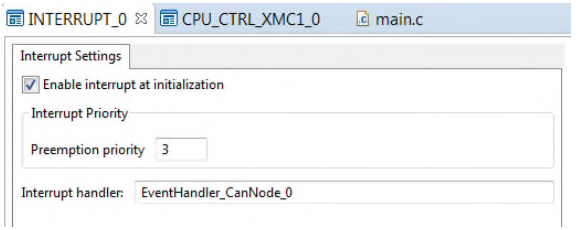
- Click  to add new APP
- Double-click **INTERRUPT** APP and close window



- Open APP configuration editor
  - In **APP Dependency** view, double-click INTERRUPT

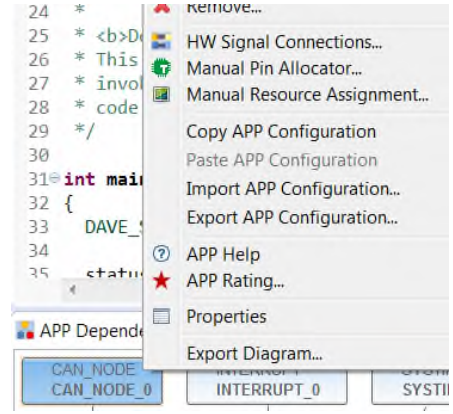


- Configure APP instance
  - In APP configuration window, configure ISR for CAN_NODE_0 as "EventHandler_CanNode_0"

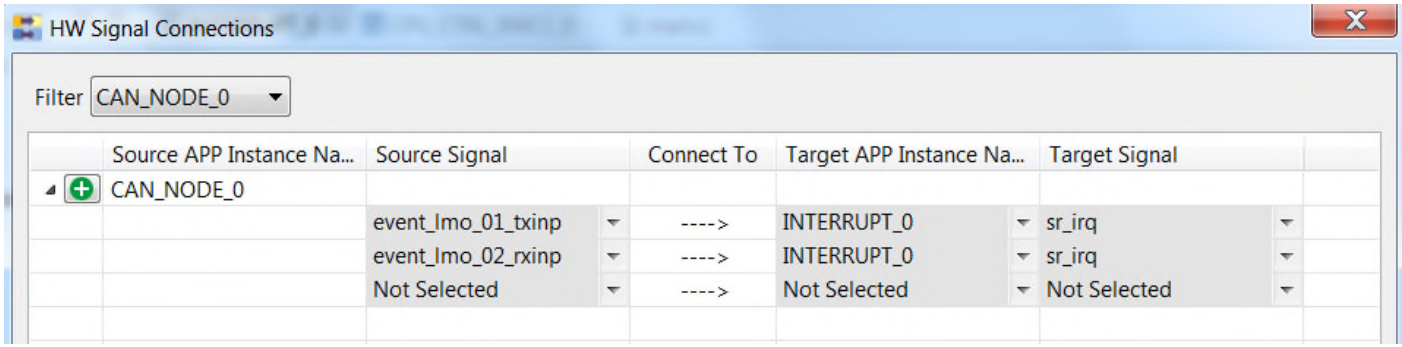


Getting Started – Example – Blinky based on DAVE™ APPs (11/13)

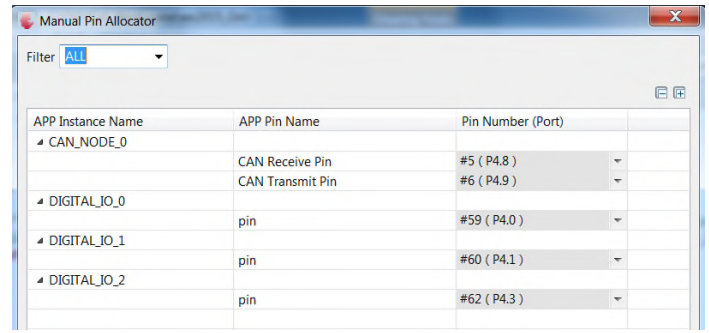
- Right click on the CAN_NODE_0 APP and select HW signal connection



- Select MO transmit event and link it to NVIC node. Then, click  and link MO received event to NVIC node.



- Assign CAN TX pin to P4.9 and CAN RX pin to P4.8
  - Click  to open **Manual Pin Allocator**
  - Set the desired pins as shown on the right
  - **Save and closed**



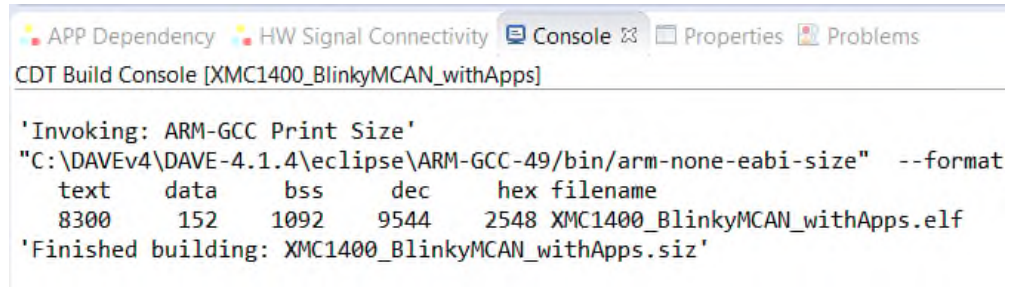
Getting Started – Example – Blinky based on DAVE™ APPs (12/13)

- Add the Interrupt Service Routine for CAN_Node_0 to main.c

```
// This ISR is the event handler for the CAN_NODE_0
void EventHandler_CanNode_0()
{
 volatile uint32_t status = 0x00;
 status = CAN_NODE_MO_GetStatus((void*)CAN_NODE_0.lmobj_ptr[1]);
 // Check receive pending status in LMO_02
 if ( status & XMC_CAN_MO_STATUS_RX_PENDING)  //XMC_CAN_MO_STATUS_NEW_DATA
 {
   // Clear the flag
   CAN_NODE_MO_ClearStatus((void*)CAN_NODE_0.lmobj_ptr[1],XMC_CAN_MO_RESET_STATUS_RX_PENDING);
   // Read the received Message object and stores the received data in the MO structure.
   CAN_NODE_MO_Receive((void*)CAN_NODE_0.lmobj_ptr[1]);
   DIGITAL_IO_ToggleOutput(&DIGITAL_IO_2);    // Toggle LED Pin 4.3 to indicating reception of data frames is done.
 }
 // Check transmit pending status in LMO_01
 status = CAN_NODE_MO_GetStatus((void*)CAN_NODE_0.lmobj_ptr[0]);
 if (status &  XMC_CAN_MO_STATUS_TX_PENDING)
 {
  // Clear the flag
  CAN_NODE_MO_ClearStatus((void*)CAN_NODE_0.lmobj_ptr[0],XMC_CAN_MO_RESET_STATUS_TX_PENDING);
 }
}
```

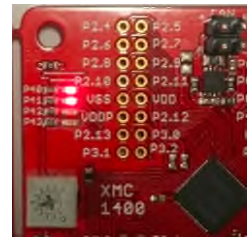
Getting Started – Example – Blinky based on DAVE™ APPs (13/13)

- Generate code
  1. Click 
- Build project
  1. Click 
  2. Wait for Build to finish



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

- LEDs P40, P41 blinks every 0.5s



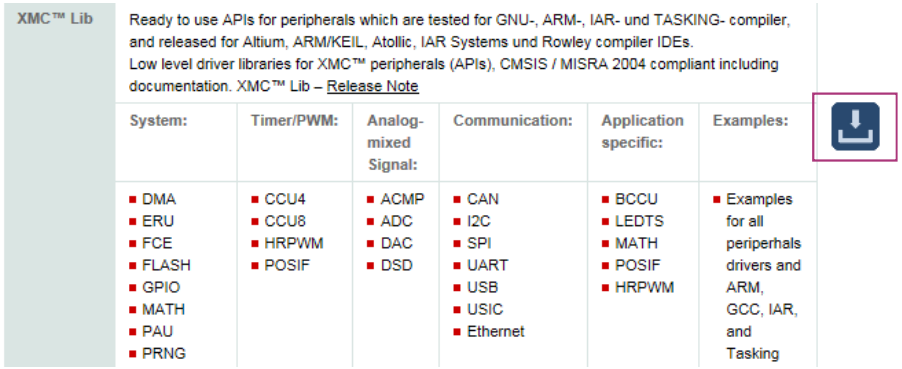
Home

References – Where to find XMC Lib documentation?

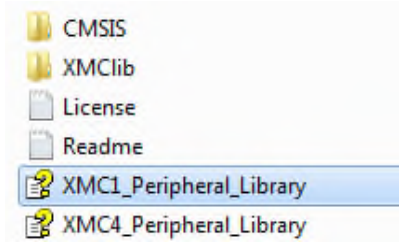
1. Go to DAVE™ Version 4 website

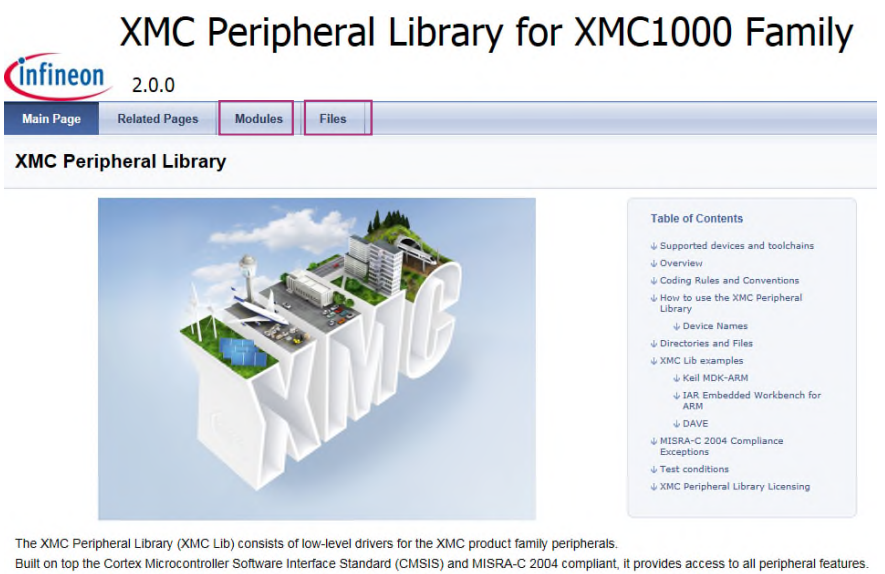
   http://www.infineon.com/dave/v4

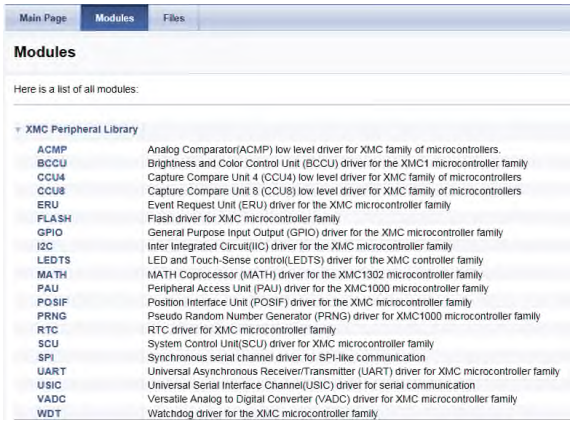
2. Download XMC Lib and unzip file

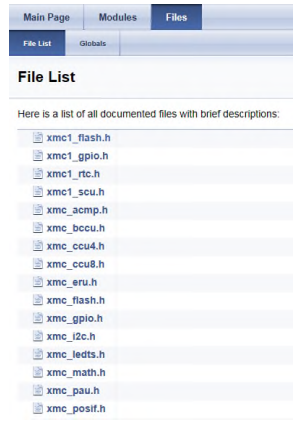


3. Open XMC1_Peripheral_Library









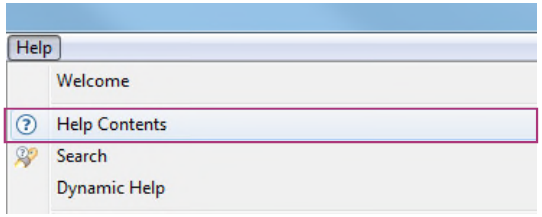
Home

Resource Listing

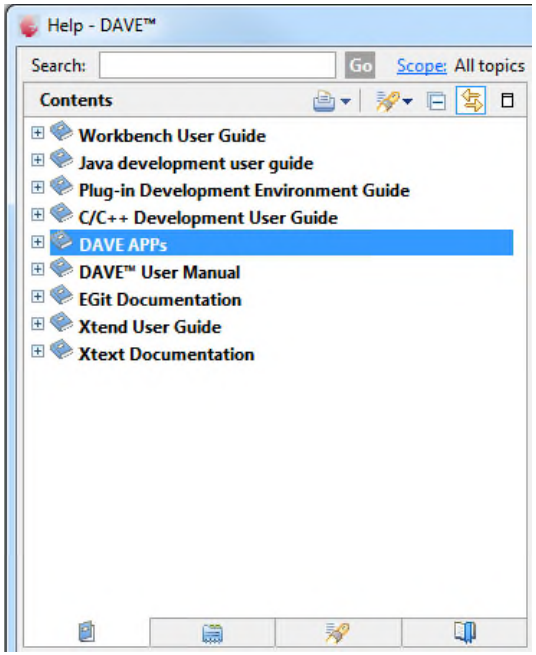
- Kit documentation:
  - Boot Kit XMC1400

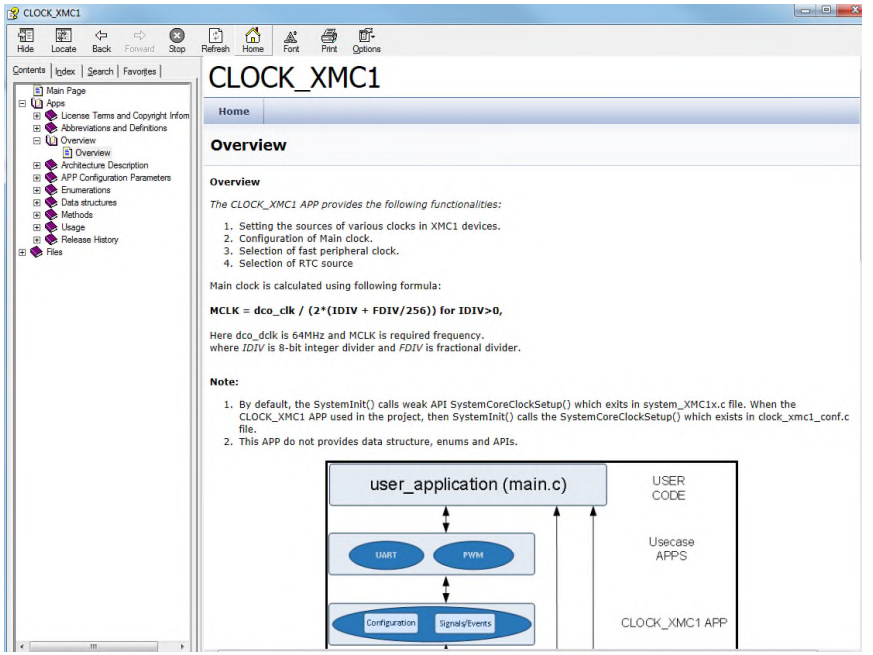
References –
Where to find App Documentation?

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



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





Home

References – Where to download Example Projects?

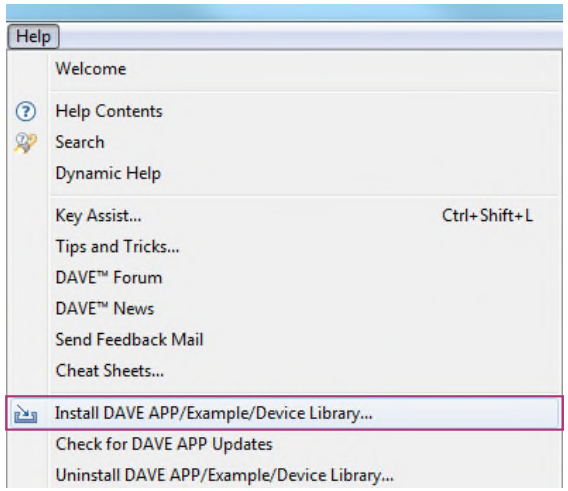
1. Example Project library within DAVE™
2. DAVE™ website
3. Example from XMC Lib package

Home

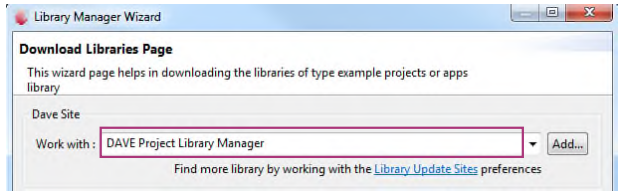
References – How to load Example Project in DAVE™? (1/4)

- Example Project library within DAVE™

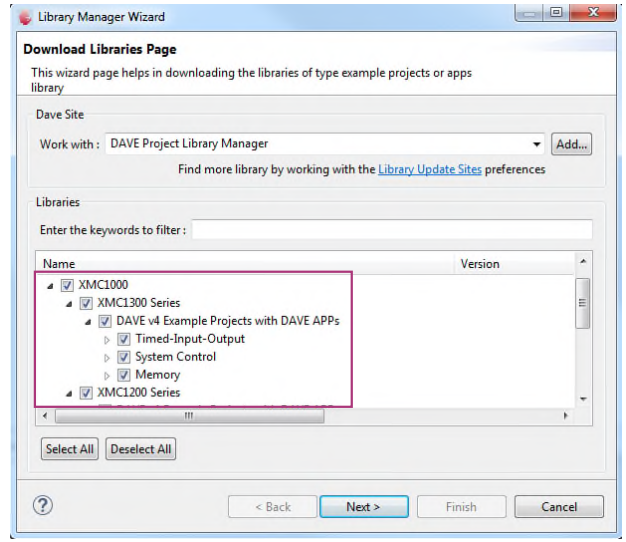
1. Help → Install DAVE APP/Example/Device Library



2. Select DAVE Project Library Manager



3. Select Examples in the Libraries window → Click Next

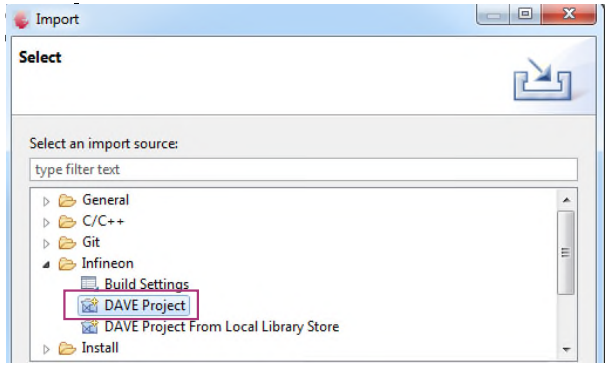


4. Accept terms of the license agreements → Click Finish

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

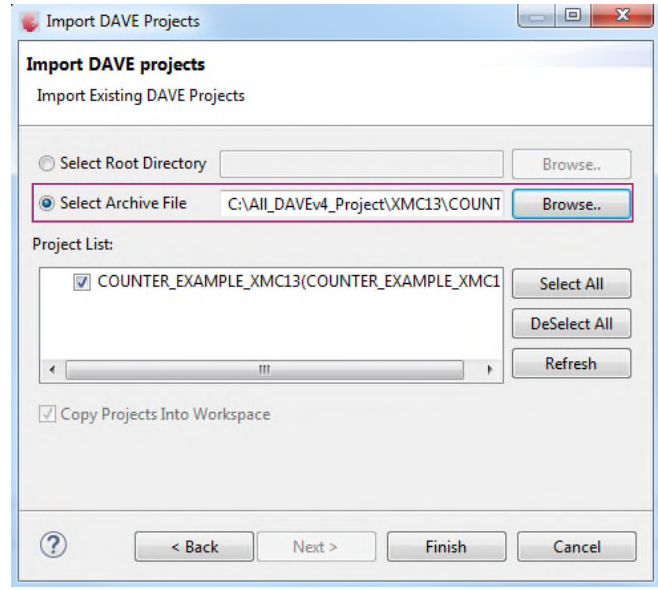
- DAVE™ website
  1. Go to DAVE™ Version 4 website

     http://www.infineon.com/dave/v4
  2. Download DAVE™ EXAMPLES

  3. In DAVE™, go to File → Import
  4. 

Next

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



  6. Click Finish

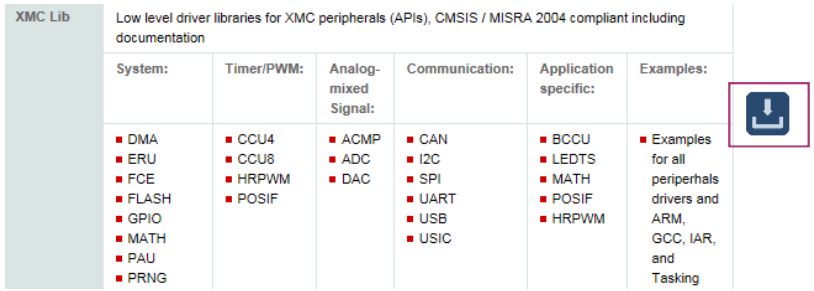
References – How to load Example Project in DAVE™? (3/4)

- Example from XMC Lib package

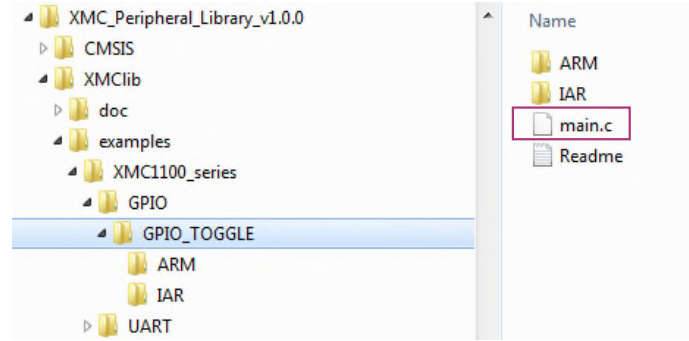
1. Go to DAVE™ Version 4 website

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

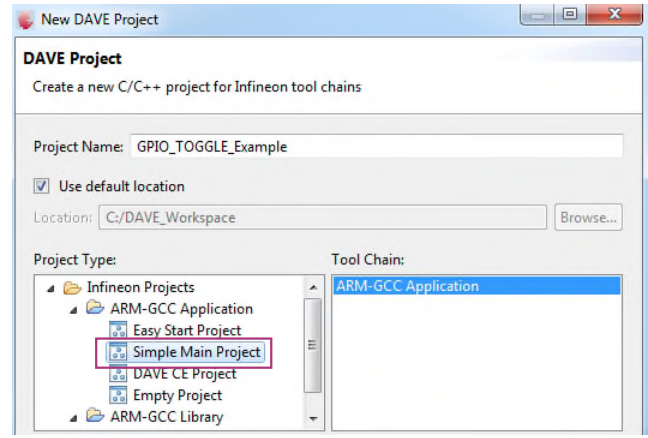
2. Download XMC Lib and unzip file



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

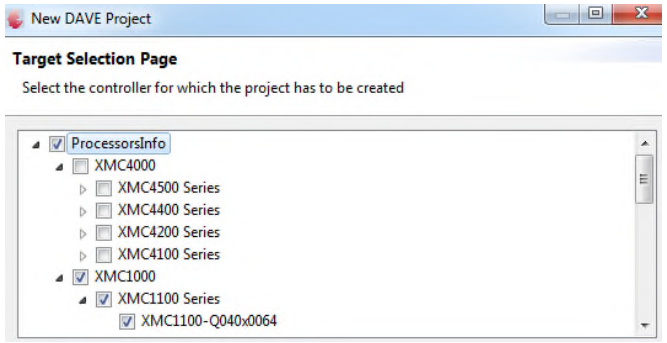


4. Create new “Simple Main Project” in DAVE™



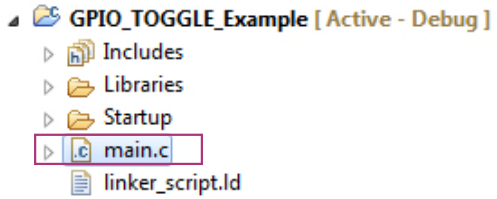
References – How to load Example Project in DAVE™? (4/4)

5. Select target device of selected main.c example



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

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

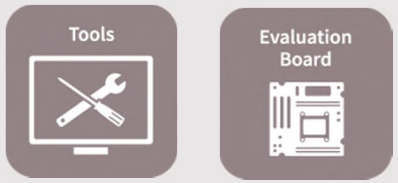


8. Click  to Build project

9. Click  to download and run project on target board

Home

Support material:

Collaterals and Brochures 	› Product Briefs › Selection Guides › Application Brochures › Presentations › Press Releases, Ads	› www.infineon.com/XMC
Technical Material 	› Application Notes › Technical Articles › Simulation Models › Datasheets, MCDS Files › PCB Design Data	› 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

ADC	Analog Digital Converter
DAVE™	Free development IDE for XMC
MO	Message Object
PWM	Pulse Width Modulation

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.