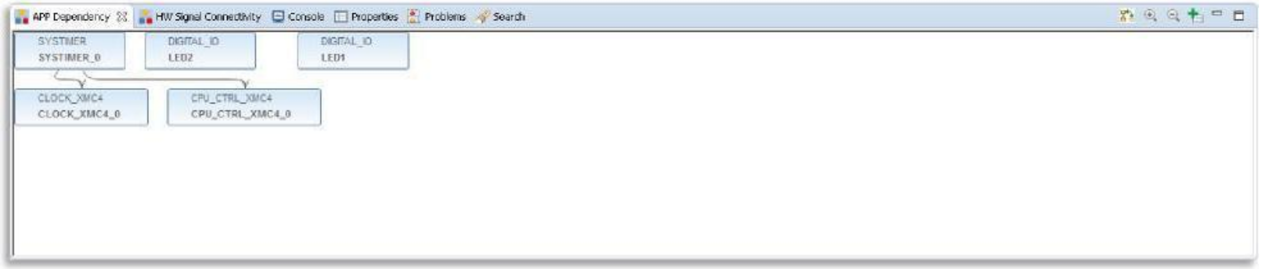
EXAMPLE NAME: XMC4500_Relax_Kit_SYSTIMER

OVERVIEW:
This example shows the usage of the SYSTIMER APP. The SYSTIMER APP uses the SysTick interrupt to call user functions periodically at a specified rate or after a given period of time expires.

DESCRIPTION:
In this example, the SYSTIMER APP is used to generate a tick interrupt every 1ms.
This time base is used to create two timers:

- a periodic timer that toggles LED1 in the board every 1s after timer is started
- a one shot timer that switches on the LED2 in the board 5s after timer is started

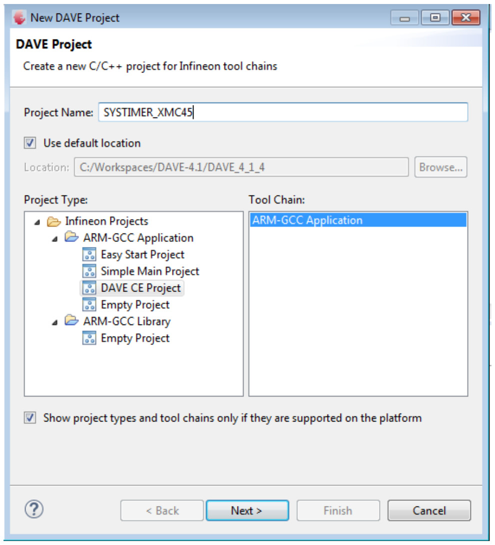


REQUIRMENTS:
Boards Required: XMC4500 Relex Kit Board (Order Nr: KIT_XMC45_RELAX_V1)

HOW TO CREATE THE PROJECT:
1. Create a new DAVE CE Project

To create new projects go to File menu, select New and choose DAVE Project.
In the New DAVE Project window:

- In the Project name: field enter the name of the new project, i.e. XMC4500_Relax_Kit_SYSTIMER
- In the Project type: section select DAVE CE Project.
- In the Toolchains: section select ARM-GCC Application
- click the Next > button



Project Name: SYSTIMER_XMC45

☑ Use default location

Location: C:/Workspaces/DAVE-4.1/DAVE_4_1_4

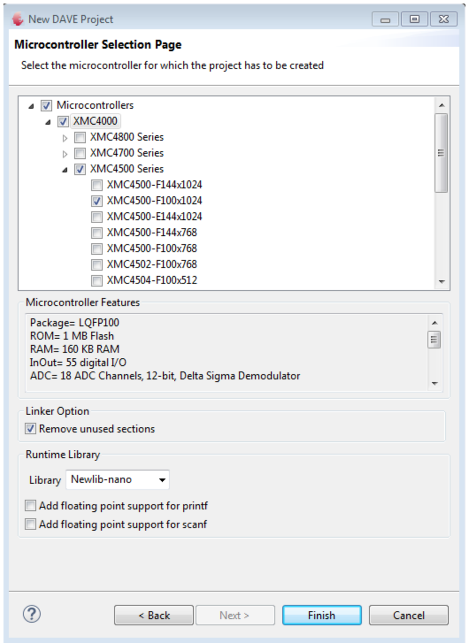
Project Type: DAVE CE Project

Tool Chain: ARM-GCC Application

☑ Show project types and tool chains only if they are supported on the platform

In the Target Selection Page:

- Select the target device, i.e. XMC4500-F100x1024
- Click the Finish button



New DAVE Project

Microcontroller Selection Page

Select the microcontroller for which the project has to be created

- ☑ Microcontrollers
  - ☑ XMC4000
    - □ XMC4800 Series
    - □ XMC4700 Series
    - ☑ XMC4500 Series
      - □ XMC4500-F144x1024
      - ☑ XMC4500-F100x1024
      - □ XMC4500-E144x1024
      - □ XMC4500-F144x768
      - □ XMC4500-F100x768
      - □ XMC4502-F100x768
      - □ XMC4504-F100x512

Microcontroller Features

Package= LQFP100
ROM= 1 MB Flash
RAM= 160 KB RAM
InOut= 55 digital I/O
ADC= 18 ADC Channels, 12-bit, Delta Sigma Demodulator

Linker Option

☑ Remove unused sections

Runtime Library

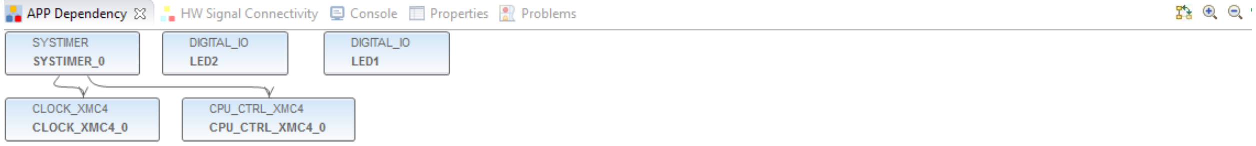
Library: Newlib-nano

□ Add floating point support for printf
□ Add floating point support for scanf

2. Add APPs to your project
Use the *Add New APP* button in toolbar or go to DAVE menu and select *Add New APP*. In the Add New APP dialog, add the following APPs by double clicking:

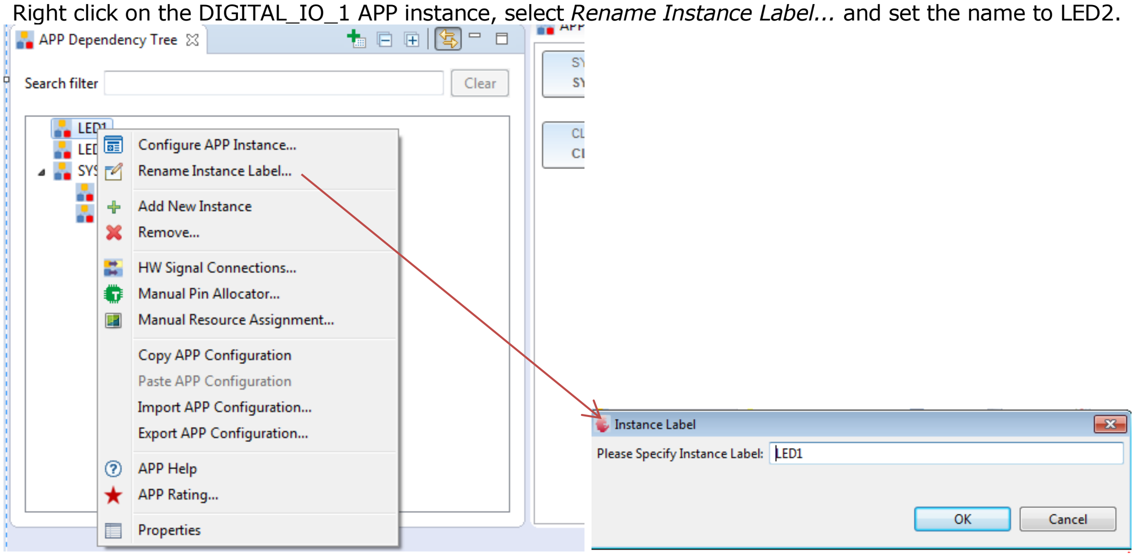
- add a SYSTIMER APP
- add two DIGITAL_IO APPs
- click the Close button

APP Dependecy View :



3. Assign names to APP instances

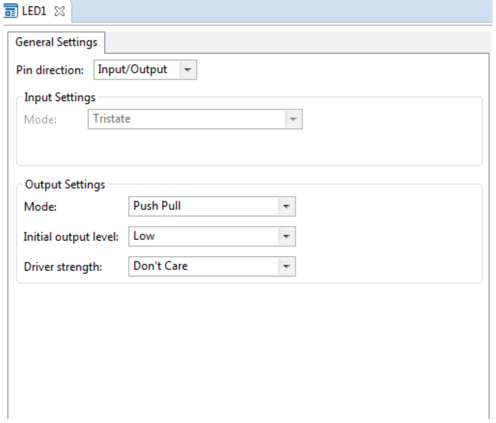
- Right click on the DIGITAL_IO_0 APP instance, select *Rename Instance Label...* and set the name to LED1.
- Right click on the DIGITAL_IO_1 APP instance, select *Rename Instance Label...* and set the name to LED2.



Please Specify Instance Label: LED1

4. Configure APPs and signals

- Double click on LED1 APP instance to configure the APP:
  - select *Input/Output* in *Pin Direction:* selection
  - keep default for the other of configuration settings.



Pin direction: Input/Output

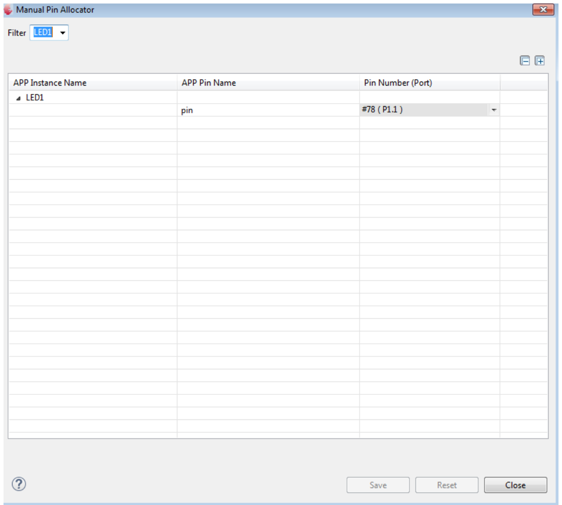
Mode: Tristate

Mode: Push Pull

Initial output level: Low

Driver strength: Don't Care

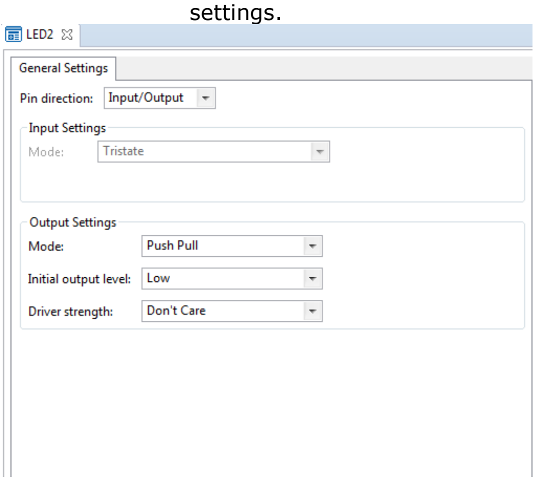
- Right click on LED1 APP instance, select *Manual Pin Allocator...* In the Manual Pin Allocator dialog:
  - Select P1.1 in the *Pin Number (Port)* selection
  - Click on Save button and finally on Close button.



Filter: LED1

APP Instance Name	APP Pin Name	Pin Number (Port)
LED1		
	pin	#78 (P1.1)

- Double click on LED2 APP instance to configure the APP:
  - Select Input/Output in *Pin Direction:* selection
  - Keep default for the other of configuration settings.



Pin direction: Input/Output

Input Settings

Mode: Tristate

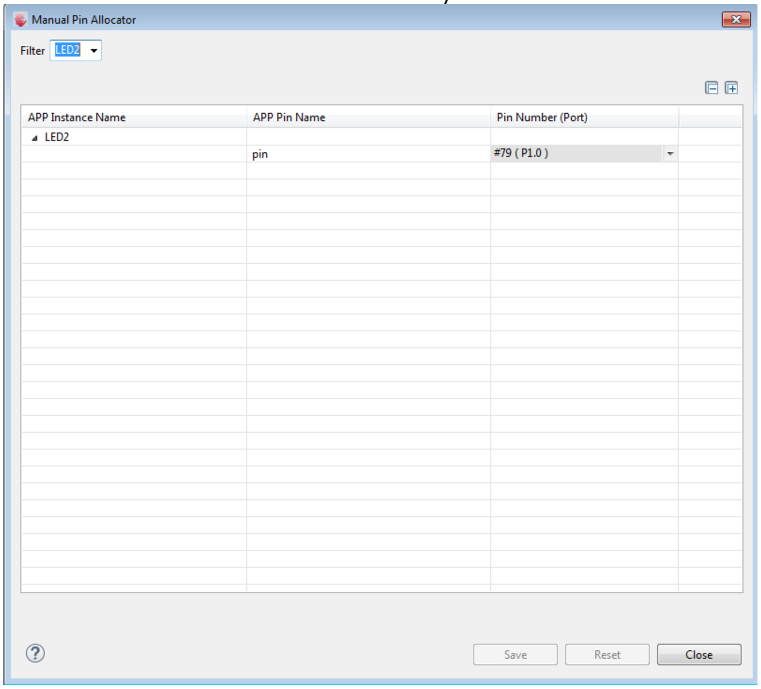
Output Settings

Mode: Push Pull

Initial output level: Low

Driver strength: Don't Care

- Right click on LED2 APP instance, select *Manual Pin Allocator...*
  In the Manual Pin Allocator dialog:
    - select P1.0 in the *Pin Number (Port)* selection
    - click on Save button and finally on Close button.

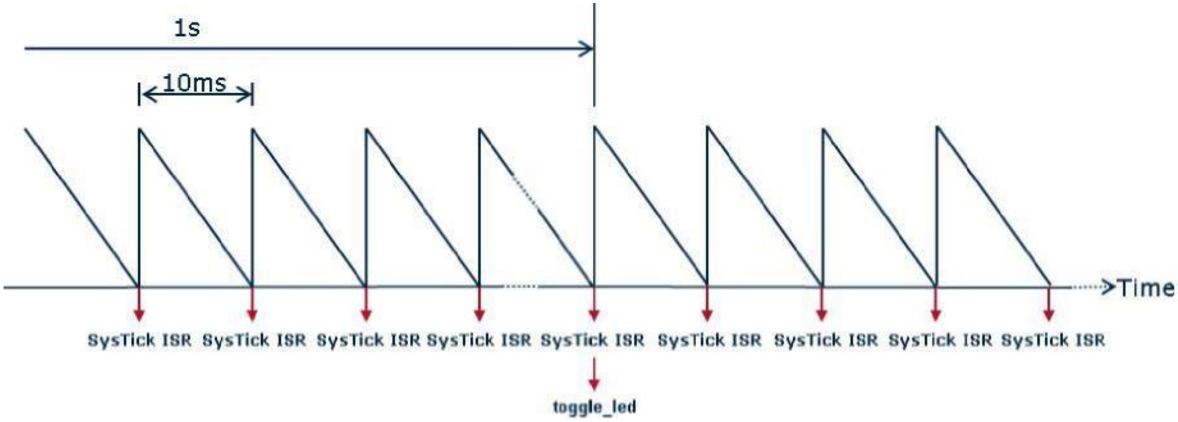


Manual Pin Allocator

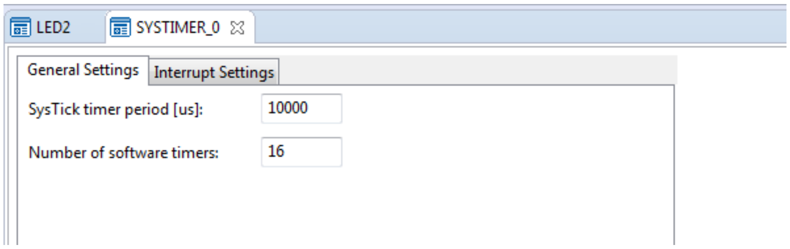
Filter LED2

APP Instance Name	APP Pin Name	Pin Number (Port)
LED2		
	pin	#79 (P1.0)

- Double click on SYSTIMER_0 APP instance to configure the APP:
    - enter 10000 us in the *System timer tick interval[us]:* input box.
      The tick interrupt will happen every 10ms.
      The tick interval is the time measurement resolution.



- enter 16 in the *Number of software timer:* input box.
  In this example 2 timers are used, but we could use up to 16 simultaneously.



SysTick timer period [us]: 10000

Number of software timers: 16

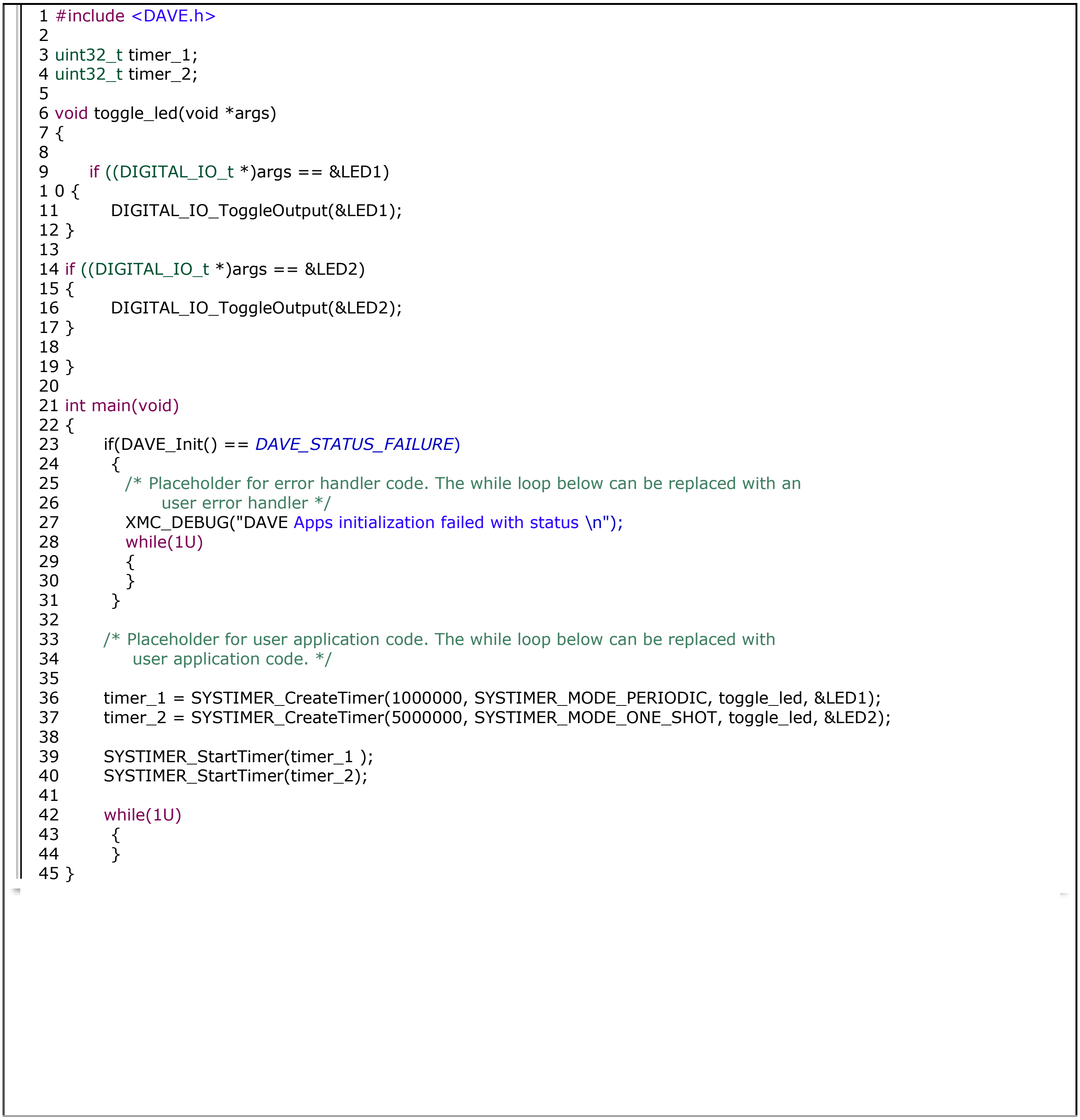
4. Generate the code

   Use the button in toolbar or go to DAVE menu and select *Generate Code.*
   In the project a new folder *Generated* is created under *DAVE* folder. This folder contains a subfolder for every APP type used in the project. Each APP subfolder contains the generated APP configuration in the form of configuration structures, i.e. SYSTIMER_conf.c. In addition, the APP API library files are copied.

5. Add user code to main.c
   (36-37) Two timers are created:

   - timer_1 periodic timer at 1s rate, triggers call to toggle_led with LED1 as parameter
   - timer_2 one shot timer expiring at 5s, triggers call to toggle_led with LED2 as parameter

   (39-40) Start the timers.
   (6-19) Add toggle_led implementation.
   In this example, depending on the value of the parameter ,with which the callback function is called, either LED1 or LED2 is toggled.



```
#include <DAVE.h>

uint32_t timer_1;
uint32_t timer_2;

void toggle_led(void *args)
{

      if ((DIGITAL_IO_t *)args == &LED1)
{
         DIGITAL_IO_ToggleOutput(&LED1);
}

if ((DIGITAL_IO_t *)args == &LED2)
{
         DIGITAL_IO_ToggleOutput(&LED2);
}

}

int main(void)
{
      if(DAVE_Init() == DAVE_STATUS_FAILURE)
       {
         /* Placeholder for error handler code. The while loop below can be replaced with an
              user error handler */
         XMC_DEBUG("DAVE Apps initialization failed with status \n");
         while(1U)
         {
         }
       }

      /* Placeholder for user application code. The while loop below can be replaced with
          user application code. */

      timer_1 = SYSTIMER_CreateTimer(1000000, SYSTIMER_MODE_PERIODIC, toggle_led, &LED1);
      timer_2 = SYSTIMER_CreateTimer(5000000, SYSTIMER_MODE_ONE_SHOT, toggle_led, &LED2);

      SYSTIMER_StartTimer(timer_1 );
      SYSTIMER_StartTimer(timer_2);

      while(1U)
       {
       }
}
```

7. Build and download to the microcontroller.