RT-Thread Nano Porting

Example Code Introduction for 32-bit NuMicro® Family

Document Information

Application	This example, based on the GPIO_OutputInput demo in BSP, shows how to port RT-Thread Nano and create 2 threads.
BSP Version	M251_M252_M254_M256_M258_Series_BSP_CMSIS_V3.02.006
Hardware	NuMaker-M252SD V1.2

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller and microprocessor based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1. Overview

This example, based on the GPIO_OutputInput demo in BSP, shows how to port RT-Thread Nano and create 2 threads. Thread 1 can control LED flashing every 500 ms through GPIO. Thread 2 alternately prints the words "Hello" and "World" every 1000 ms through UART.

1.1 Principle

RT-Thread Nano is a refined hard real-time kernel designed for systems with limited memory resources. It includes functions such as thread management, inter-thread synchronization and communication, clock management, interrupt management, and memory management, which make RT-Thread Nano ideal for embedded systems requiring high-performance real-time control. It supports a variety of CPU architectures, including Arm Cortex M0/M23/M3/M4/M7 and RISC-V, and requires minimal memory resources, with a minimum of 3 KB Flash and 1.2 KB RAM.

This example is based on the M252SD2AE microcontroller, which includes 64 KB Flash and 12 KB RAM, and shows how to port the RT-Thread Nano and create 2 threads.

1.2 Demo Result

After the sample code is compiled and programmed, the LED (PB14) flashes every 500 ms, as shown in Figure 1-1.

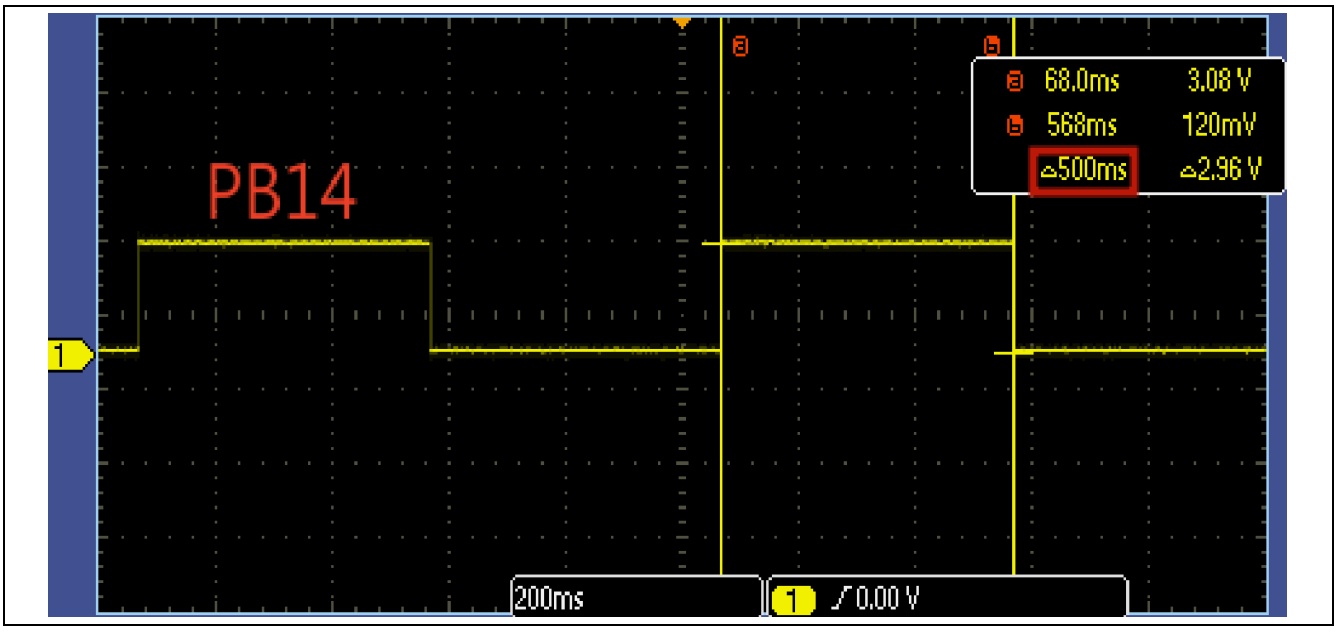


Figure 1-1 Compilation and Programming Results

The message printed by the serial port is shown in Figure 1-2.

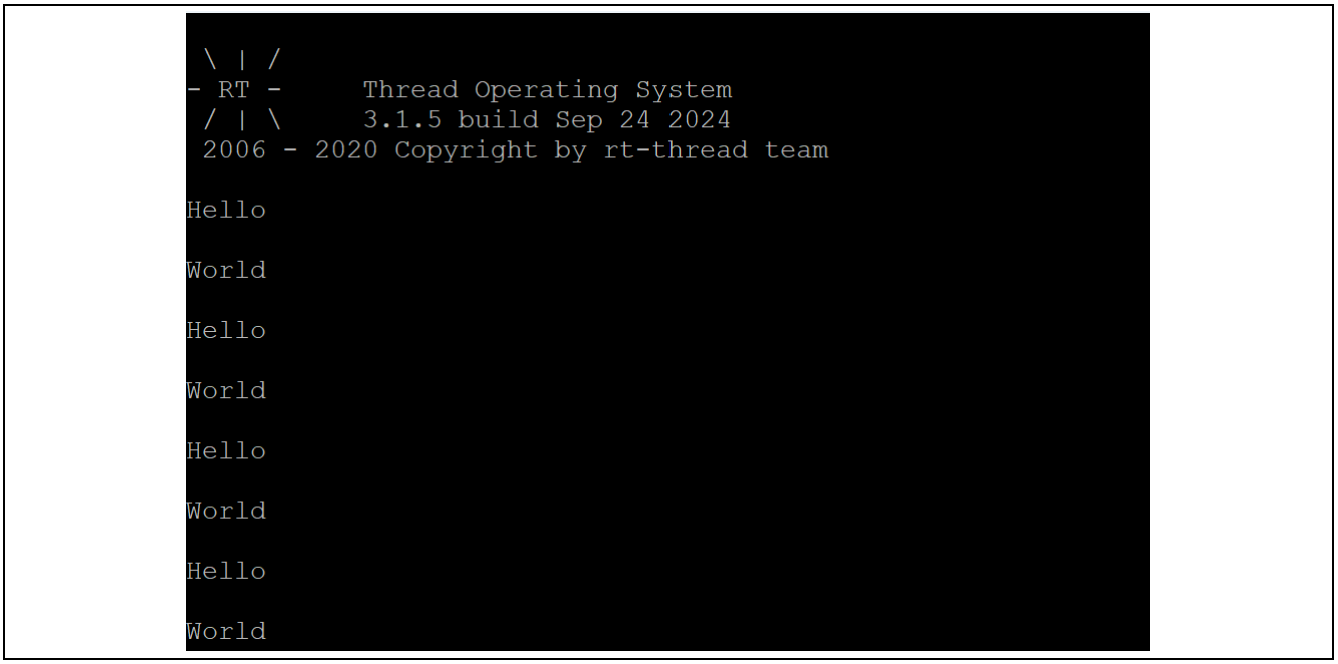


Figure 1-2 Serial Port Prints a Message

2. Code Description

2.1 Download RT-Thread Nano Source Code

Click the link below to download the RT-Thread Nano source code on the official website.

https://www.rt-thread.org/document/site/#/rt-thread-version/rt-thread-nano/an0038-nano-introduction

The source code is extracted as follows.

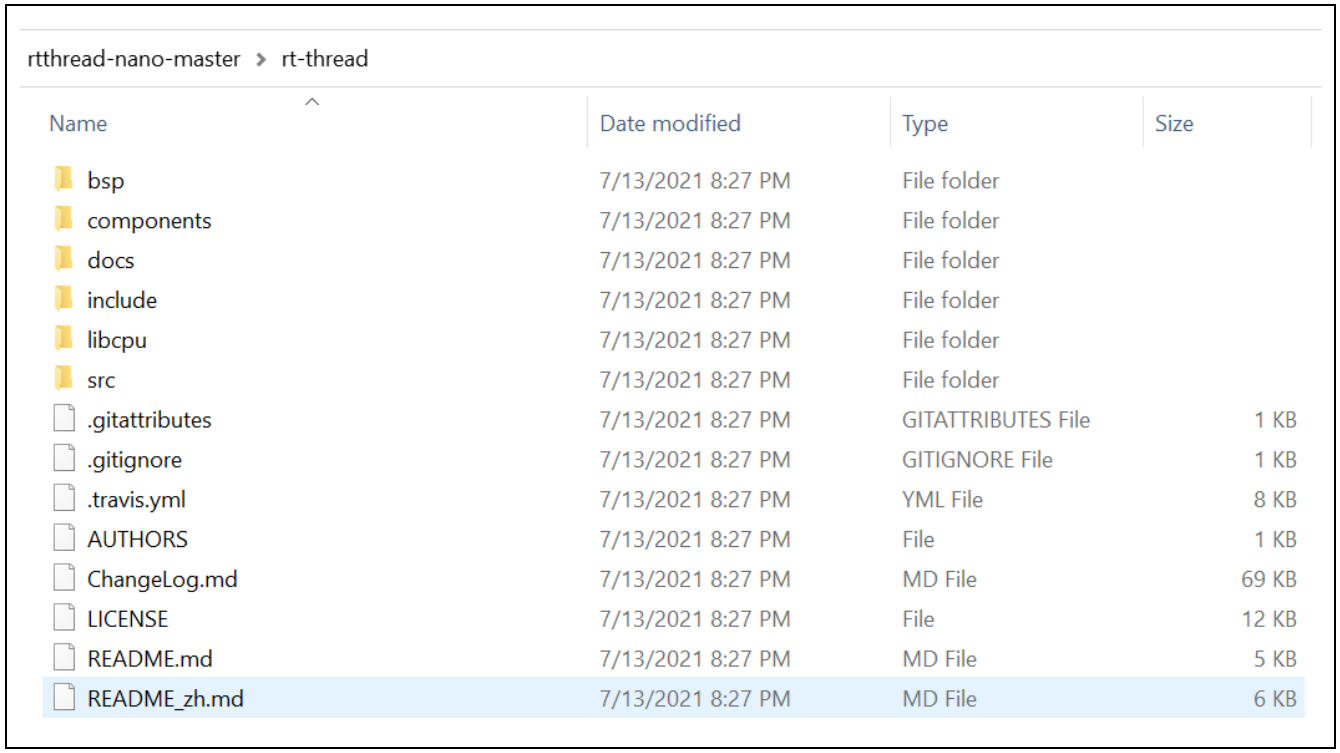


Figure 2-1 rt-thread File Structure

2.2 Copy rt-thread File to BSP Path

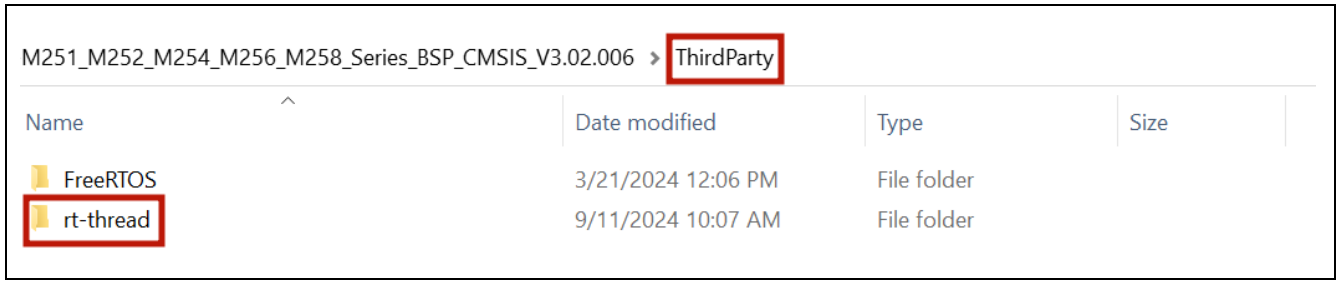


Figure 2-2 rt-thread Path

2.3 Create a New "rt-thread" Group

Create a new "rt-thread" group in the project and add the associated source code files.

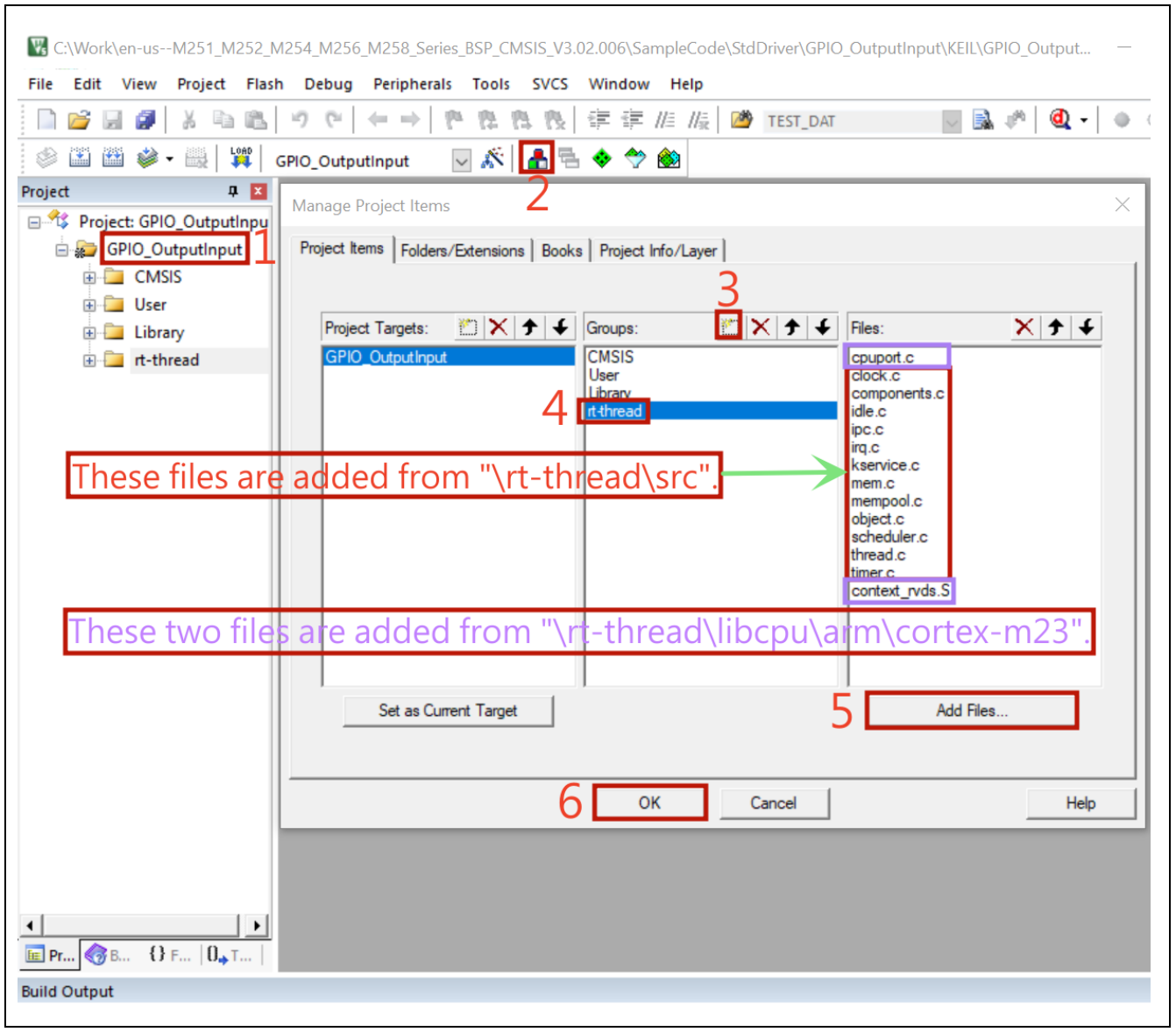


Figure 2-3 Create a New "rt-thread" Group and Add Source Code Files

2.4 Copy *Board.c* and *rtconfig.h* Files

Copy the *Board.c* and *rtconfig.h* files from the "\rt-thread\bsp_template\" path to the "\GPIO_OutputInput\" path, then add them to the "User" group.

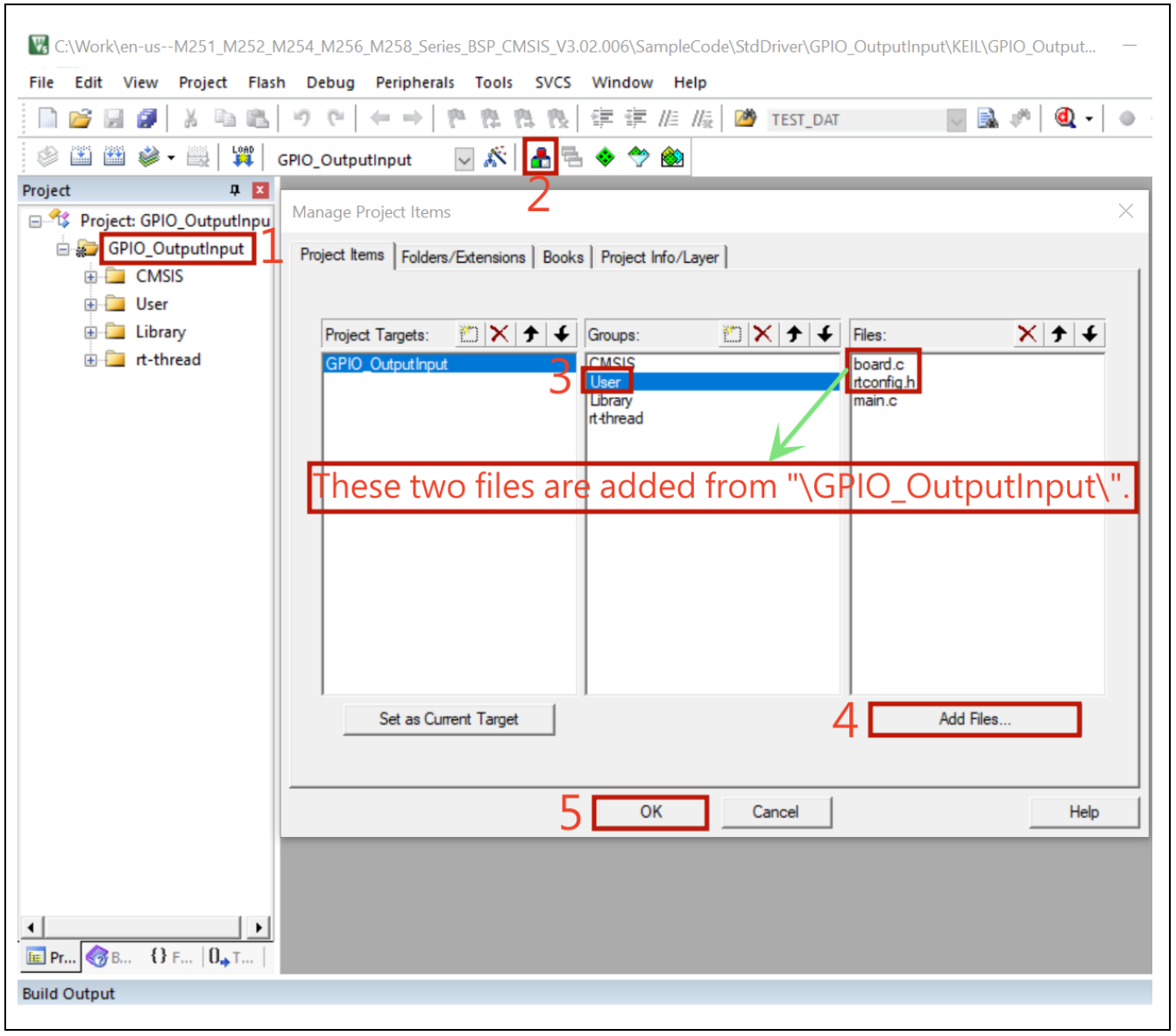


Figure 2-4 Add Files to the "User" Group

2.5 Modify "Include Paths"

Add rt-thread file include path in project options.

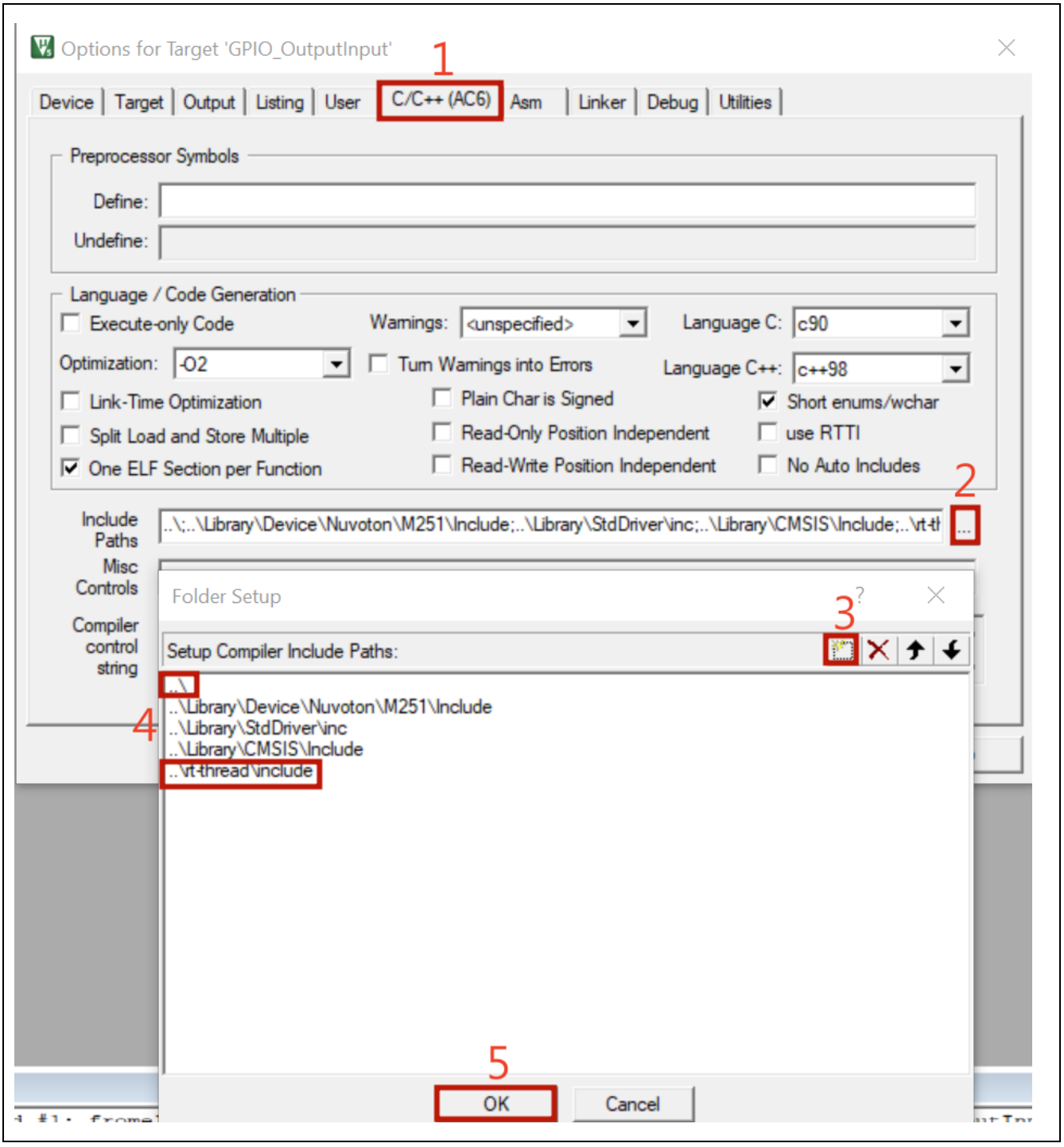


Figure 2-5 Add the rt-thread File Include Paths

2.6 Modify *board.c* File

Include header files *NuMicro.h* and *stdio.h*.

```
#include <rthw.h>
#include <rtthread.h>
#include <stdio.h>
#include "NuMicro.h"
```

Because the maximum SRAM size of the target chip is 12 KB, it is necessary to modify "# define RT_HEAP_SIZE (15 * 1024) " to "# define RT_HEAP_SIZE (4 * 1024) ".

```
/*
 * Please modify RT_HEAP_SIZE if you enable RT_USING_HEAP
 * the RT_HEAP_SIZE max value = (sram size - ZI size), 1024 means 1024 bytes
 */
#define RT_HEAP_SIZE (4*1024)
```

The original array variable and functions are redefined as follows to avoid non-word-aligned risks.

```
static rt_uint32_t rt_heap[RT_HEAP_SIZE/sizeof(rt_uint32_t)];

RT_WEAK void *rt_heap_begin_get(void)
{
    return (void *)rt_heap;
}

RT_WEAK void *rt_heap_end_get(void)
{
    return (void *)((rt_uint32_t)rt_heap + sizeof(rt_heap));
}
```

Add the SysTick_Handler function under the rt_os_tick_callback function.

```
   void SysTick_Handler(void)
   {
    rt_os_tick_callback();
   }
```

In the rt_hw_board_init function, comment the first line, then add SYS_Init, SystemCoreClockUpdate, and SysTick_Config functions.

```
    //#error "TODO 1: OS Tick Configuration."
    /*
     * TODO 1: OS Tick Configuration
     * Enable the hardware timer and call the rt_os_tick_callback function
     * periodically with the frequency RT_TICK_PER_SECOND.
     */
    /* System Init */
    SYS_UnlockReg();
    SYS_Init();
    SYS_LockReg();

    /* System Clock Update */
    SystemCoreClockUpdate();

    /* System Tick Configuration */
    SysTick_Config(SystemCoreClock / RT_TICK_PER_SECOND);
```

Copy the SYS_Init function from the *main.c* file to the *board.c* file.

In the uart_init function, comment the first line, then add the UART reset function and the baud rate setting function.

```
static int uart_init(void)
{
    //#error "TODO 2: Enable the hardware uart and config baudrate."
    /* Reset UART module */
    SYS_ResetModule(UART0_RST);

    /* Configure UART0 and set UART0 baud rate */
    UART_Open(UART0, 115200);

    return 0;
}
```

In the rt_hw_console_output function, comment the first line and then write the printf function.

```
void rt_hw_console_output(const char *str)
{
    //#error "TODO 3: Output the string 'str' through the uart."
    printf("%s", str);
}
```

2.7 Modify *main.c* File

Include header file *rtthread.h*.

```
    #include "stdio.h"
    #include "NuMicro.h"
    #include <rtthread.h>
```

Define all thread-related variables.

```
    static struct rt_thread blink_thread;
    static char blink_thread_stack[256];
    static struct rt_thread print_thread;
    static char print_thread_stack[256];
```

Thread 1 function design.

```
    void blink_thread_entry(void *parameter)
    {
        while (1)
        {
            PB14 = 1;
            rt_thread_mdelay(500);
            PB14 = 0;
            rt_thread_mdelay(500);
        }
    }
```

Thread 2 function design.

```
    void print_thread_entry(void *parameter)
    {
        while (1)
        {
            rt_kprintf("\nHello\n");
            rt_thread_mdelay(1000);
            rt_kprintf("\nWorld\n");
```

```
            rt_thread_mdelay(1000);
        }
    }
```

Delete the SYS_Init and UART0_Init functions.

Re-write the main function.

```
int main(void)
{
    rt_err_t ret;

    /* Configure PB.14 as Output mode */
    GPIO_SetMode(PB, BIT14, GPIO_MODE_OUTPUT);

    /* Create thread 1 */
    ret = rt_thread_init(&blink_thread,
                         "led blink",
                         blink_thread_entry,
                         RT_NULL,
                         &blink_thread_stack[0],
                         sizeof(blink_thread_stack),
                         RT_THREAD_PRIORITY_MAX-2,
                         20);

    if (ret == RT_EOK)
    {
        /* After the thread 1 is created successfully, start the thread 1 */
        rt_thread_startup(&blink_thread);
    }

    /* Create thread 2*/
    ret = rt_thread_init(&print_thread,
                         "print info",
                         print_thread_entry,
                         RT_NULL,
                         &print_thread_stack[0],
                         sizeof(print_thread_stack),
                         RT_THREAD_PRIORITY_MAX-3,
                         20);

    if (ret == RT_EOK)
    {
        /* After the thread 2 is created successfully, start the thread 2 */
        rt_thread_startup(&print_thread);
    }

    return 0;
}
```

2.8 Modify *rtconfig.h* File

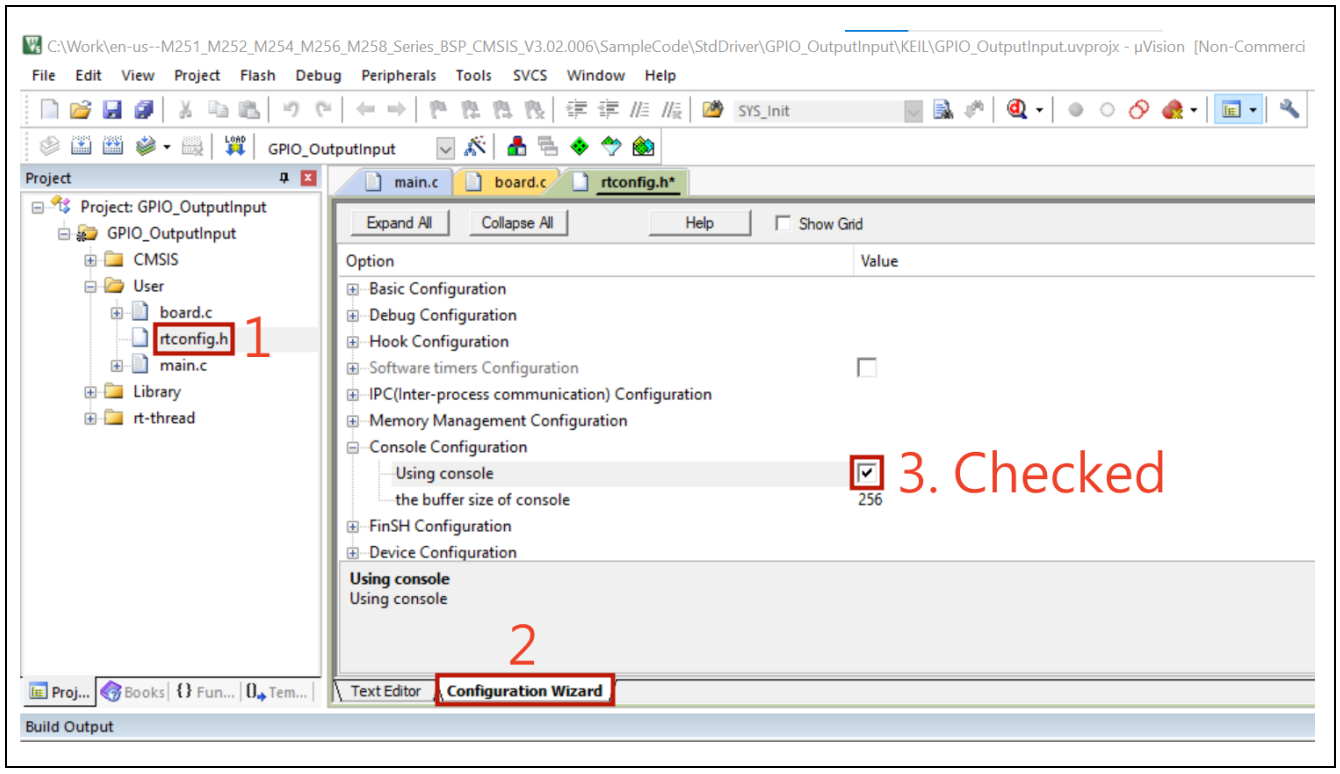
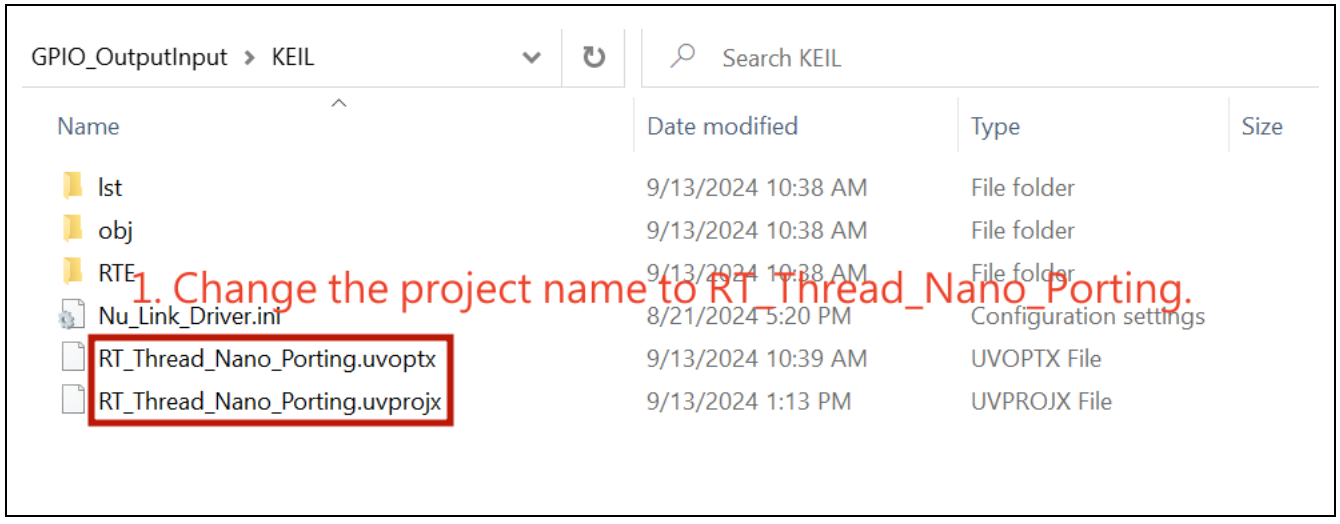


Figure 2-6 Configure "console"

2.9 Modify Project Name

Modify the project name to "RT_Thread_Nano_Porting".



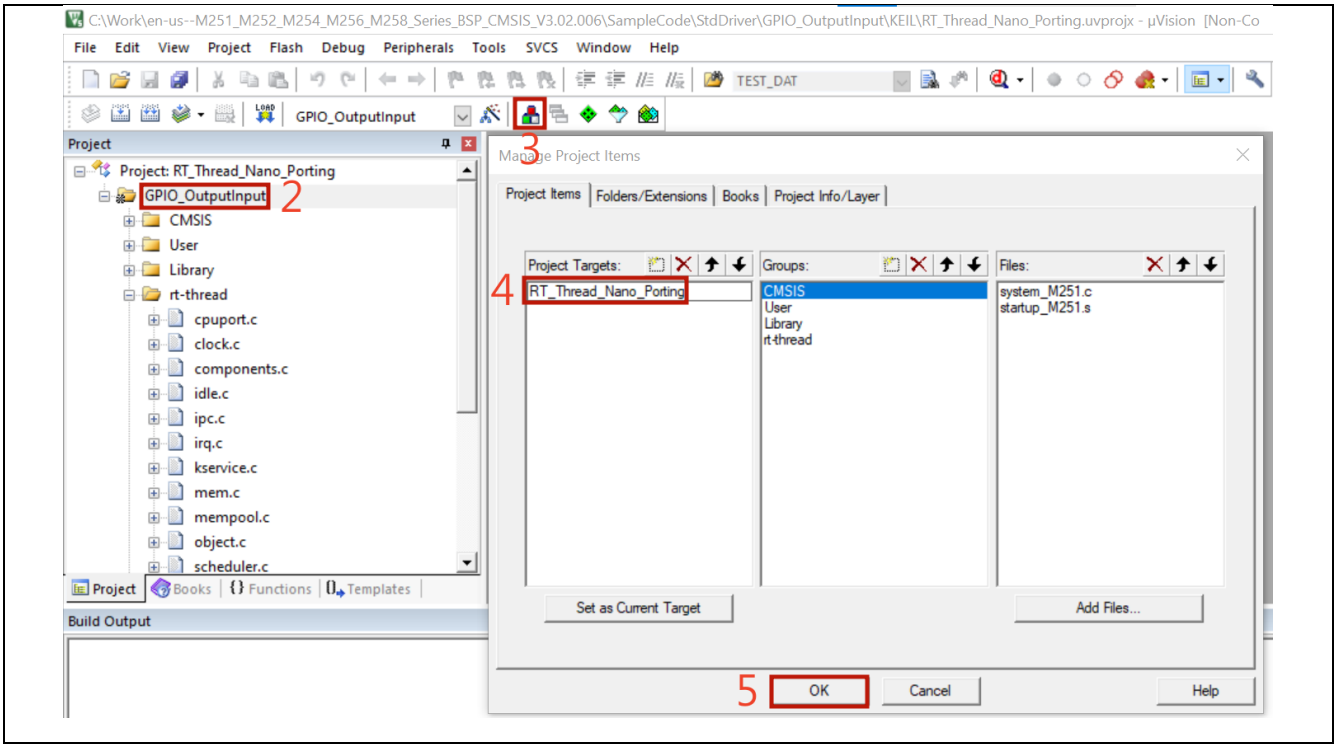


Figure 2-7 Modify Project Name to "RT_Thread_Nano_Porting"

2.10 Modify Related File Name

Modify the file names accordingly and delete unnecessary files.

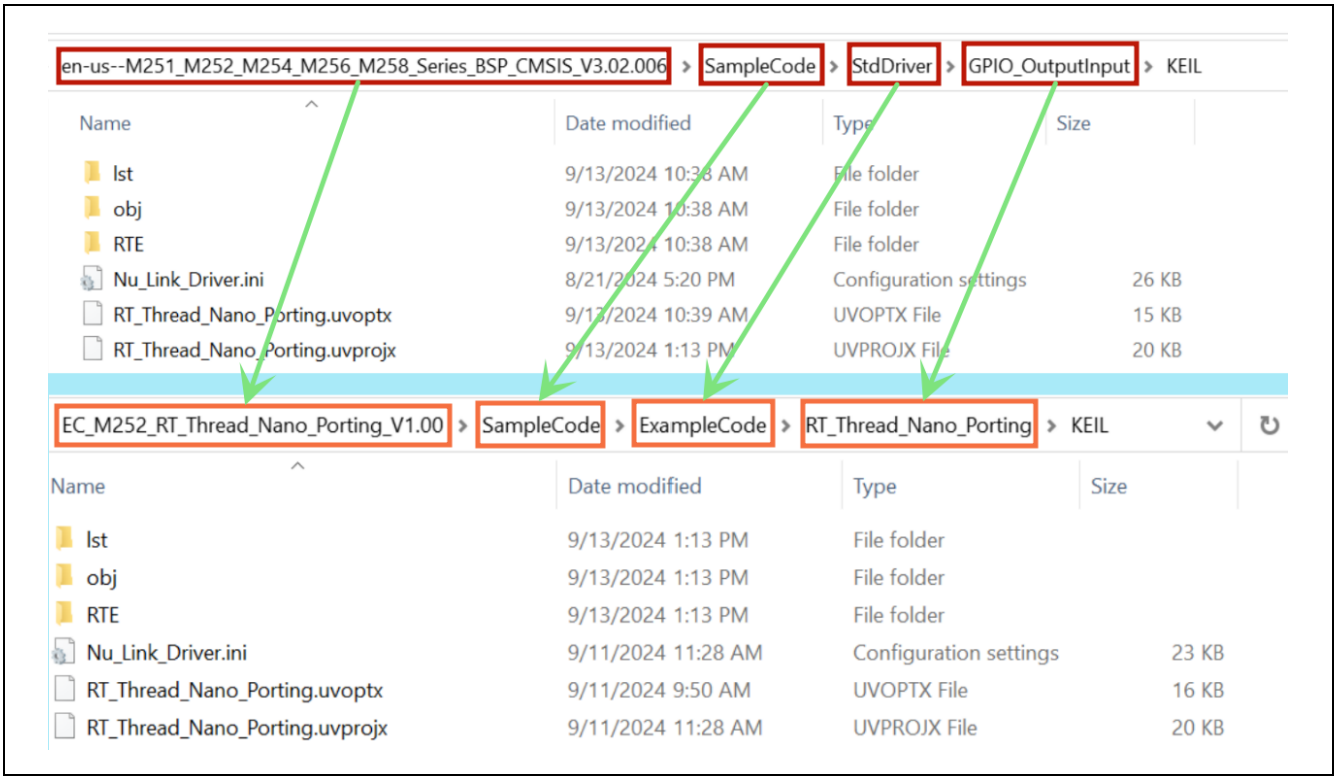


Figure 2-8 Modify the Name of Each File

3. Software and Hardware Requirements

3.1 Software Requirements

- BSP version
  - M251_M252_M254_M256_M258_Series_BSP_CMSIS_V3.02.006
- IDE version
  - Keil uVersion 5.36

3.2 Hardware Requirements

- Circuit components
  - NuMaker-M252SD V1.2
- Pin Connect
  - You need to enable the USB-to-serial port function by setting the DIP switch on the Nu-Link2-Me.

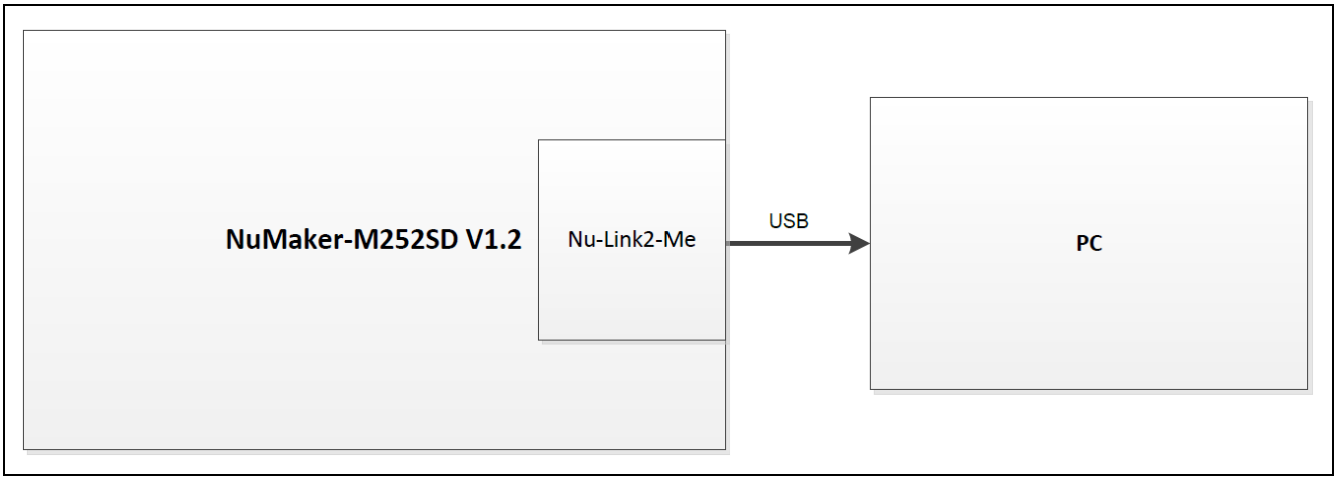


Figure 3-1 Pin Connect

4. Directory Information

EC_M252_RT_Thread_Nano_Porting_V1.00	
Library	Sample code header and source files
CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
ThirdParty	
rt-thread	Source file of RT-Thread Nano code
SampleCode	
ExampleCode	
Project	Source file of example code

Figure 4-1 Directory Structure

5. Example Code Execution

1. Browse the sample code folder as described in the Directory Information section and double-click *RT_Thread_Nano_Porting.uvprojx*.
2. Enter Keil compile mode.
   - Build
   - Download
   - Start/Stop debug session
3. Enter debug mode.
   - Run

6. Revision History

Date	Revision	Description
2024.10.08	1.00	Initial version.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.