

# CM2003 Series CMSIS BSP Guide

Directory Introduction for 32-bit NuMicro® Family

## Directory Information

<b>Document</b>	Driver reference guide and revision history.
<b>Library</b>	Driver header and source files.
<b>SampleCode</b>	Driver sample code.

*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 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](http://www.nuvoton.com)

## TABLE OF CONTENTS

1	DOCUMENT.....	3
2	LIBRARY .....	4
3	SAMPLECODE .....	5
4	SAMPLECODE\ISP .....	6
5	SAMPLECODE\POWERMANAGEMENT .....	7
6	SAMPLECODE\STDDRIVER.....	8
	System Manager (SYS) .....	8
	Flash Memory Controller (FMC) .....	8
	General Purpose I/O (GPIO).....	8
	Timer Controller (TIMER).....	9
	Watchdog Timer (WDT) .....	9
	Window Watchdog Timer (WWDT) .....	10
	Pulse Width Modulation Controller (PWM) .....	10
	UART Interface Controller (UART).....	10
	I <sup>2</sup> C Serial Interface Controller (I <sup>2</sup> C) .....	11
	Universal Serial Control Interface Controller – UART Mode (USCI-UART) .....	11
	Universal Serial Control Interface Controller – SPI Mode (USCI-SPI) .....	12
	Universal Serial Control Interface Controller – I <sup>2</sup> C Mode (USCI-I2C) .....	12
	Analog-to-Digital Converter (ADC) .....	13
	Enhanced Input Capture Timer (ECAP) .....	13

## 1 Document

<b>CMSIS.html</b>	Document of CMSIS version 6.1.0.
<b>NuMicro CM2003 Driver Reference Guide.chm</b>	This document describes the usage of drivers in CM2003 BSP.
<b>NuMicro CM2003 Series CMSIS BSP Revision History.pdf</b>	This document shows the revision history of CM2003 BSP.

## 2 Library

<b>CMSIS</b>	Cortex® Microcontroller Software Interface Standard (CMSIS) V6.1.0 definitions by Arm® Corp.
<b>Device</b>	CMSIS compliant device header file.
<b>StdDriver</b>	All peripheral driver header and source files.

### 3 SampleCode

<b>Hard_Fault_Sample</b>	<p>Show hard fault information when hard fault happened.</p> <p>The hard fault handler shows some information included program counter, which is the address where the processor was executing when the hard fault occurs. The listing file (or map file) can show what function and instruction that was.</p> <p>It also shows the Link Register (LR), which contains the return address of the last function call. It can show the status where CPU comes from to get to this point.</p>
<b>ISP</b>	Sample codes for In-System-Programming.
<b>PowerManagement</b>	Sample codes for power management.
<b>Semihost</b>	Show how to print and get character through IDE console window.
<b>StdDriver</b>	Sample code to demonstrate the usage of CM2003 series MCU peripheral driver APIs.
<b>Template</b>	A project template for CM2003 series MCU.

## 4 SampleCode\ISP

ISP_I2C	In-System-Programming Sample code through I <sup>2</sup> C interface.
ISP_RS485	In-System-Programming Sample code through RS485 interface.
ISP_UART	In-System-Programming Sample code through UART interface.

## **5 SampleCode\PowerManagement**

**SYS\_PowerDown\_MinCurrent**

Demonstrate how to minimize power consumption when entering power down mode.

## 6 SampleCode\StdDriver

### System Manager (SYS)

<b>SYS_BODWakeup</b>	Demonstrate how to wake up system from Power-down mode by brown-out detector interrupt.
----------------------	---

### Flash Memory Controller (FMC)

<b>FMC_CRC32</b>	Demonstrate how to use FMC CRC32 ISP command to calculate the CRC32 checksum of APROM and LDROM.
<b>FMC_ExecInSRAM</b>	Implement a code and execute it in SRAM to program embedded Flash.
<b>FMC_IAP</b>	Demonstrate FMC IAP boot mode and show how to use vector remap function. LDROM image was embedded in APROM image and be programmed to LDROM Flash at run-time. This sample also shows how to branch between APROM and LDROM.
<b>FMC_MultiBoot</b>	Implement a multi-boot system to boot from different applications in APROM or LDROM by VECMAP.
<b>FMC_ReadAllOne</b>	Demonstrate how to use FMC Read-All-One ISP command to verify APROM or LDROM pages are all 0xFFFFFFFF or not.
<b>FMC_RW</b>	Show FMC read Flash IDs, erase, read, and write functions.

### General Purpose I/O (GPIO)

<b>GPIO_EINTAndDebounce</b>	Show the usage of GPIO external interrupt function and de-bounce function.
<b>GPIO_INT</b>	Show the usage of GPIO interrupt function.
<b>GPIO_OutputInput</b>	Show how to set GPIO pin mode and use pin data input and output control.
<b>GPIO_PowerDown</b>	Show how to wake up system from Power-down mode

by GPIO interrupt.

## Timer Controller (TIMER)

<b>TIMER_CaptureCounter</b>	Show how to use the Timer capture function to capture Timer counter value.
<b>TIMER_Delay</b>	Demonstrate the usage of TIMER_Delay API to generate a 1 second delay.
<b>TIMER_EventCounter</b>	Use TM0 pin to demonstrate Timer event counter function.
<b>TIMER_FreeCountingMode</b>	Use the timer TM0_EXT pin to demonstrate timer free counting mode function. And displays the measured input frequency to UART console.
<b>TIMER_InterTimerTriggerMode</b>	Use the timer TM0 pin to demonstrate inter timer trigger mode function. Also display the measured input frequency to UART console.
<b>TIMER_Periodic</b>	Use the Timer periodic mode to generate Timer interrupt every 1 second.
<b>TIMER_PeriodicINT</b>	Implement Timer counting in periodic mode.
<b>TIMER_SW_RTC</b>	This sample code performs how to use software to simulate RTC. In power down mode, using the Timer to wake-up the MCU and add RTC value per second.
<b>TIMER_TimeoutWakeup</b>	Use timer to wake up system from Power-down mode periodically.
<b>TIMER_ToggleOut</b>	Demonstrate the Timer0 toggle out function on TM0 pin.

## Watchdog Timer (WDT)

<b>WDT_TimeoutWakeupAndReset</b>	Implement WDT time-out interrupt event to wake up system and generate time-out reset system event while WDT time-out reset delay period expired.
----------------------------------	--

## Window Watchdog Timer (WWDT)

WWDT_ReloadCounter	Show how to reload the WWDT counter value.
--------------------	--

## Pulse Width Modulation Controller (PWM)

PWM_240KHz_SwitchDuty	Demonstrate how to set PWM0 channel 0 output 240 kHz waveform and switch duty in each 0.5%.
PWM_Brake	Demonstrate how to use PWM brake function.
PWM_Capture	Capture the PWM Channel 2 waveform by PWM Channel 0.
PWM_DeadTime	Demonstrate how to use PWM Dead Time function.
PWM_DoubleBuffer	Change duty cycle and period of output waveform by PWM double buffer function.
PWM_OutputWaveform	Demonstrate how to use PWM counter output waveform.
PWM_SwitchDuty	Change duty cycle of output waveform by configured period.
PWM_SyncStart	Demonstrate how to use PWM counter synchronous start function.

## UART Interface Controller (UART)

UART_AutoBaudRate	Show how to use auto baud rate detection function.
UART_AutoFlow	Transmit and receive data using auto flow control.
UART_IrDA	Transmit and receive UART data in UART IrDA mode.
UART_RS485	Transmit and receive data in UART RS485 mode.
UART_SingleWire	Transmit and receive data in UART single-wire mode.
UART_TxRxFunction	Transmit and receive data from PC terminal through RS232 interface.
UART_Wakeup	Show how to wake up system from Power-down mode

by UART interrupt.

## I<sup>2</sup>C Serial Interface Controller (I<sup>2</sup>C)

<b>I2C_Double_Buffer_Slave</b>	Demonstrate how to set I <sup>2</sup> C two-level buffer in Slave mode to receive 256 bytes data from a master. This sample code needs to work with I2C_MultiBytes_Master.
<b>I2C_EEPROM</b>	Show how to use I <sup>2</sup> C interface to access EEPROM.
<b>I2C_Master</b>	Show how a master accesses a slave. This sample code needs to work with I2C_Slave.
<b>I2C_MultiBytes_Master</b>	Show how to set I <sup>2</sup> C Multi bytes API Read and Write data to Slave. This sample code needs to work with I2C_Slave.
<b>I2C_SingleByte_Master</b>	Show how to use I <sup>2</sup> C Single byte API Read and Write data to Slave. This sample code needs to work with I2C_Slave.
<b>I2C_Slave</b>	Demonstrate how to set I <sup>2</sup> C in Slave mode to receive 256 bytes data from a master. This sample code needs to work with I2C_Master.
<b>I2C_Wakeup_Slave</b>	Show how to wake up MCU from Power-down mode via the I <sup>2</sup> C interface. This sample code needs to work with I2C_Master.

## Universal Serial Control Interface Controller – UART Mode (USCI-UART)

<b>USCI_UART_AutoBaudRate</b>	Show how to use auto baud rate detection function.
<b>USCI_UART_Autoflow</b>	Transmit and receive data using auto flow control.
<b>USCI_UART_RS485</b>	Transmit and receive data in RS485 mode.
<b>USCI_UART_TxRxFunction</b>	Transmit and receive data from PC terminal through a RS232 interface.
<b>USCI_UART_Wakeup</b>	Show how to wake up system from Power-down mode by USCI interrupt in UART mode.

## Universal Serial Control Interface Controller – SPI Mode (USCI-SPI)

<b>USCI_SPI_MasterMode</b>	Configure USCI_SPI0 as Master mode and demonstrate how to communicate with an off-chip SPI Slave device. This sample code needs to work with USCI_SPI_SlaveMode sample code.
<b>USCI_SPI_SlaveMode</b>	Configure USCI_SPI0 as Slave mode and demonstrate how to communicate with an off-chip SPI Master device. This sample code needs to work with USCI_SPI_MasterMode sample code.

## Universal Serial Control Interface Controller – I<sup>2</sup>C Mode (USCI-I2C)

<b>USCI_I2C_EEPROM</b>	Demonstrate how to access EEPROM through a USCI_I2C interface.
<b>USCI_I2C_Master</b>	Demonstrate how a Master accesses Slave. This sample code needs to work with USCI_I2C_Slave sample code.
<b>USCI_I2C_Master_10bit</b>	Demonstrate how a Master uses 10-bit addressing access Slave. This sample code needs to work with USCI_I2C_Slave_10bit sample code.
<b>USCI_I2C_MultiBytes_Master</b>	Demonstrate how to use multi-bytes API to access slave. This sample code needs to work with USCI_I2C_Slave sample code.
<b>USCI_I2C_SingleByte_Master</b>	Demonstrate how to use single byte API to access slave. This sample code needs to work with USCI_I2C_Slave sample code.
<b>USCI_I2C_Slave</b>	Demonstrate how to set USCI_I2C in slave mode to receive the data from a Master. This sample code needs to work with USCI_I2C_Master sample code.
<b>USCI_I2C_Slave_10bit</b>	Demonstrate how to set USCI_I2C in 10-bit addressing slave mode to receive the data from a Master. This sample code needs to work with USCI_I2C_Master_10bit sample code.
<b>USCI_I2C_Wakeup_Slave</b>	Demonstrate how to set USCI_I2C to wake up MCU from Power-down mode. This sample code needs to

work with USCI\_I2C\_Master sample code.

## Analog-to-Digital Converter (ADC)

<b>ADC_ADINT_Trigger</b>	Use ADINT interrupt to do the ADC Single-cycle scan conversion.
<b>ADC_BandGap</b>	Convert Band-gap and print conversion result.
<b>ADC_BandGapCalculateAVDD</b>	Demonstrate how to calculate analog voltage (AVdd) by using band-gap.
<b>ADC_BurstMode</b>	Perform A/D Conversion with ADC burst mode.
<b>ADC_ContinuousScanMode</b>	Perform A/D Conversion with ADC continuous scan mode.
<b>ADC_PWM_Trigger</b>	Demonstrate how to trigger ADC by PWM.
<b>ADC_ResultMonitor</b>	Monitor the conversion result of channel 2 by the digital compare function.
<b>ADC_SingleCycleScanMode</b>	Perform A/D Conversion with ADC single cycle scan mode.
<b>ADC_SingleMode</b>	Perform A/D Conversion with ADC single mode.
<b>ADC_STADC_Trigger</b>	Show how to trigger ADC by STADC pin.
<b>ADC_SwTrg_Trigger</b>	Trigger ADC by writing ADC software trigger register.
<b>ADC_Timer_Trigger</b>	Show how to trigger ADC by Timer.

## Enhanced Input Capture Timer (ECAP)

<b>ECAP_GetInputFreq</b>	Show how to use ECAP interface to get input frequency.
--------------------------	--



### **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.*