M471 Series CMSIS BSP Guide

Directory Introduction for 32-bit NuMicro® Family

Directory Information

Document	Driver reference guide and revision history.
Library	Driver header and source files.
SampleCode	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

TABLE OF CONTENTS

1 Document

CMSIS.html	Document of CMSIS version 5.1.1.
NuMicro M471 Series CMSIS BSP Driver Reference Guide.chm	This document describes the usage of drivers in M471 BSP.
NuMicro M471 Series CMSIS BSP Revision History.pdf	This document shows the revision history of M471 BSP.

2 Library

CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) V5.1.1 definitions by Arm® Corp.
Device	CMSIS compliant device header file.
StdDriver	All peripheral driver header and source files.

3 SampleCode

CortexM4	Cortex®-M4 sample code.
Hard_Fault_Sample	Show hard fault information when hard fault happened. The hard fault handler shows some information including program counter, which is the address where the processor is executing when the hard fault occurs. The listed file (or map file) can show what function and instruction that was. 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.
ISP	Sample codes for In-System-Programming.
Semihost	Show how to print and get character through IDE console window.
StdDriver	Sample code to demonstrate the usage of M471 series MCU peripheral driver APIs.
Template	A project template for M471 series MCU.
XOM	Demonstrate how to create XOM library and use it.

4 SampleCode\CortexM4

BitBand	Demonstrate the usage of Cortex®-M4 Bit-band.
DSP_FFT	Demonstrate how to call ARM CMSIS DSP library to calculate FFT.
MPU	Demonstrate the usage of Cortex®-M4 MPU.

5 SampleCode\ISP

ISP_I2C	In-System-Programming Sample code through an I^2C interface.
ISP_RS485	In-System-Programming Sample code through a RS485 interface.
ISP_SPI	In-System-Programming Sample code through a SPI interface.
ISP_UART	In-System-Programming Sample code through a UART interface.

6 SampleCode\StdDriver

System Manager (SYS)

SYS_BODWakeup	Demonstrate how to wake up system from Power-down mode by brown-out detector interrupt.
SYS_PLLClockOutput	Change system clock to different PLL frequency and output system clock to CLKO pin.
SYS_PowerDown_MinCurrent	Demonstrate how to minimize power consumption when entering power down mode.
SYS_TrimHIRC	Demonstrate how to use LXT to trim HIRC.

Clock Controller (CLK)

CLK_ClockDetector	Demonstrate the usage of clock fail detector and clock frequency range detector function.

Flash Memory Controller (FMC)

FMC_CRC32	Demonstrate how to use FMC CRC32 ISP command to calculate the CRC32 checksum of APROM and LDROM.
FMC_Dual_Bank	Demonstrate how dual processes work in dual bank Flash architecture.
FMC_DualBankFwUpdate	Implement a firmware update mechanism based on dual bank Flash architecture.
FMC_ExeInSRAM	Implement a code and execute in SRAM to program embedded Flash.
FMC_IAP	Demonstrate FMC IAP boot mode and show how to use vector remap function. LDROM image is embedded in APROM image and will be programmed to LDROM Flash at run-time. This sample also shows how to branch between APROM and LDROM.
FMC_MultiWordProgram	Show how to use FMC multi-word program ISP command to program APROM 0x10000~0x20000 area.

FMC_ReadAllOne	Demonstrate how to use FMC Read-All-One ISP command to verify if APROM/LDROM pages are all 0xFFFFFFFF.
FMC_RW	Show FMC read Flash IDs, erase, read, and write functions.

Data Flash Memory Controller (DFMC)

DFMC_CRC32	Demonstrate how to use DFMC CRC32 ISP command to calculate the CRC32 checksum of Data Flash.
DFMC_NonBlocking	Demonstrate how to program Data Flash in non-blocking mode.
DFMC_ReadAllOne	Demonstrate how to use DFMC Read-All-One ISP command to verify if Data Flash pages are all 0xFFFFFFFF.
DFMC_RW	Show DFMC read Flash IDs, erase, read, and write functions.

General Purpose I/O (GPIO)

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

I^{2}C Serial Interface Controller (I^{2}C)

I2C_EEPROM	Show how to use I^{2}C interface to access EEPROM.
I2C_GCMode_Master	Show how a master uses I^{2}C address 0x0 to write data to a slave. This sample code needs to work with I2C_GCMode_Slave.

I2C_GCMode_Slave	Show how a slave receives data from a master in GC (General Call) mode. This sample code needs to work with I2C_GCMode_Master.
I2C_Loopback	Demonstrate how to set I^{2}C Master mode and Slave Mode, and show how a master accesses a slave on a chip.
I2C_Master	Show how a master accesses a slave. This sample code needs to work with I2C_Slave.
I2C_MultiBytes_Master	Show how to set I^{2}C Multi bytes API Read and Write data to Slave. This sample code needs to work with I2C_Slave.
I2C_PDMA_TRX	Demonstrate I^{2}C PDMA mode and need to connect I2C0 (master) and I2C1 (slave).
I2C_SingleByte_Master	Show how to use I^{2}C Single byte API Read and Write data to Slave. This sample code needs to work with I2C_Slave.
I2C_Slave	Demonstrate how to set I^{2}C in Slave mode to receive 256 bytes data from a master. This sample code needs to work with I2C_Master.
I2C_SMBus	Show how to control SMBus interface and use SMBus protocol between Host and Slave.
I2C_Wakeup_Slave	Show how to wake up MCU from Power-down mode via the I^{2}C interface. This sample code needs to work with I2C_Master.

PDMA Controller (PDMA)

PDMA_BasicMode	Use PDMA channel 1 to transfer data from memory to memory.
PDMA_ScatterGather	Use PDMA channel 1 to transfer data from memory to memory by scatter-gather mode.
PDMA_ScatterGather_PingPongBuffer	Use PDMA to implement Ping-Pong buffer by scatter-gather mode (memory to memory).

Real Time Clock (RTC)

RTC_Alarm_Test	Demonstrate the RTC alarm function. It sets an alarm 10 seconds after execution.
RTC_Alarm_Wakeup	Use RTC alarm interrupt event to wake up system.
RTC_Time_Display	Demonstrate the RTC function and displays current time to the UART console.

Timer Controller (TIMER)

TIMER_ACMPTrigger	Use ACMP to trigger Timer0 counter reset mode.
TIMER_CaptureCounter	Show how to use the Timer capture function to capture Timer counter value.
TIMER_Delay	Demonstrate the usage of TIMER_Delay API to generate a 1 second delay.
TIMER_EventCounter	Use TM0 pin to demonstrate Timer event counter function.
TIMER_FreeCountingMode	Use the timer TM0_EXT pin to demonstrate timer free counting mode function, and display the measured input frequency to UART console.
TIMER_InterTimerTriggerMode	Use the timer TM0 pin to demonstrate inter timer trigger mode function, and display the measured input frequency to UART console.
TIMER_Periodic	Use the Timer periodic mode to generate Timer interrupt every 1 second.
TIMER_PeriodicINT	Implement Timer counting in periodic mode.
TIMER_PWM_ChangeDuty	Change duty cycle and period of output waveform in PWM up count type.
TIMER_PWM_OutputWaveform	Demonstrate output different duty waveform in Timer0~3 PWM.
TIMER_TimeoutWakeup	Use timer to wake up system from Power-down mode periodically.

TIMER_ToggleOut	Demonstrate the Timer0 toggle out function on TM0 pin.

Watchdog Timer (WDT)

WDT_TimeoutWakeupAndReset	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.

Basic PWM Generator and Capture Timer (BPWM)

BPWM_Capture	Capture the BPWM0 Channel 2 waveform by BPWM1 Channel 0.
BPWM_DoubleBuffer	Change duty cycle and period of output waveform by PWM Double Buffer function.
BPWM_SwitchDuty	Change duty cycle of output waveform by configured period.
BPWM_OutputWaveform	Demonstrate how to use BPWM counter output waveform.
BPWM_SyncStart	Demonstrate how to use BPWM counter synchronous start function.

Enhanced PWM Generator and Capture Timer (EPWM)

EPWM_AccumulatorINT_TriggerPDMA	Demonstrate how to use EPWM accumulator interrupt trigger PDMA.
EPWM_Brake	Demonstrate how to use EPWM brake function.
EPWM_Capture	Capture the EPWM1 Channel 0 waveform by EPWM1 Channel 2.
EPWM_DeadTime	Demonstrate how to use EPWM Dead Time function.

EPWM_DoubleBuffer	Change duty cycle and period of output waveform by EPWM Double Buffer function.
EPWM_OutputWaveform	Demonstrate how to use EPWM counter output waveform.
EPWM_PDMA_Capture	Capture the EPWM1 Channel 0 waveform by EPWM1 Channel 2, and use PDMA to transfer captured data.
EPWM_SwitchDuty	Change duty cycle of output waveform by configured period.
EPWM_SyncStart	Demonstrate how to use EPWM counter synchronous start function.

Serial Peripheral Interface (SPI)

SPI_Loopback	SPI read/write demo connecting SPI MISO and MOSI pins.
SPI_MasterFIFOMode	Configure SPI as Master mode and demonstrate how to communicate with an off-chip SPI slave device with FIFO mode. This sample code needs to work with SPI_SlaveFIFOMode.
SPI_PDMA_LoopTest	SPI read/write demo in PDMA mode. Connecting SPI MISO and MOSI pins. Both TX PDMA function and RX PDMA function will be enabled.
SPI_SlaveFIFOMode	Configure SPI as Slave mode and demonstrate how to communicate with an off-chip SPI master device with FIFO mode. This sample code needs to work with SPI_MasterFIFOMode.
SPII2S_Master	Configure SPI in I^{2}S Master mode and demonstrate how I^{2}S works in Master mode.
SPII2S_PDMA_NAU8822	An I^{2}S demo with PDMA function connected to audio codec NAU8822.
SPII2S_PDMA_Play	An I^{2}S demo for playing data and demonstrating how I^{2}S works with PDMA.

SPII2S_PDMA_PlayRecord	An I^2S demo for playing and recording data with PDMA function.
SPII2S_PDMA_Record	An I^2S demo for recording data and demonstrating how I^2S works with PDMA.
SPII2S_Slave	Configure SPI as I^2S Slave mode and demonstrate how I^2S works in Slave mode. This sample code needs to work with I2S_Master.

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_LIN	Demonstrate how to send data to LIN bus.
UART_PDMA	Demonstrate UART transmit and receive function with PDMA.
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.

Pseudo Random Number Generator (PRNG)

PRNG_KeyGeneration	Generate random numbers using PRNG.

Customize IR Receiver (CIR)

CIR_DataMatchWakeup	Use CIR to wake up system from Power-down mode while first 8 bits data matched.

CIR_TPWM_TRX	Demonstrate how to use CIR to receive data from an IR LED that was driven by timer PWM.
CIR_TV_Remote_Controller	Demonstrate how to use CIR APIs to convert the output signal of an IR receiver.

CRC Controller (CRC)

CRC_CCITT	Implement CRC in CRC-CCITT mode and get the CRC checksum result.
CRC_CRC32_PDMA	Implement CRC in CRC-32 mode and get the CRC checksum result.
CRC_CRC8	Implement CRC in CRC-8 mode and get the CRC checksum result.

Enhanced Analog-to-Digital Converter (EADC)

EADC_Accumulate	Demonstrate how to get accumulate conversion results.
EADC_ADINT_Trigger	Use ADINT interrupt to do the EADC continuous scan conversion.
EADC_Average	Demonstrate how to get average conversion results.
EADC_BandGap	Convert Band-gap (channel 24) and print conversion result.
EADC_BandGapCalculateAVDD	Demonstrate how to calculate battery voltage (AV_{DD}) by using band-gap.
EADC_EPWM_Trigger	Demonstrate how to trigger EADC by EPWM.
EADC_PDMA_EPWM_Trigger	Demonstrate how to trigger EADC by EPWM and transfer conversion data by PDMA.
EADC_Pending_Priority	Demonstrate how to trigger multiple sample modules and got conversion results in order of priority.
EADC_ResultMonitor	Monitor the conversion result of channel 2 by the digital compare function.

EADC_SwTrg_Trigger	Trigger EADC by writing EADC software trigger register.
EADC_TempSensor	Convert temperature sensor (channel 25) and print conversion result.
EADC_Timer_Trigger	Show how to trigger EADC by Timer.
EADC_Vref	Demonstrate how to select EADC reference voltage source and trigger delay time.

Digital-to-Analog Converter (DAC)

DAC_PDMA_TimerTrigger	Show how Timer triggers DAC to fetch data with PDMA and convert sine wave outputs.
DAC_SoftwareTrigger	Demonstrate how software triggers DAC to convert sine wave outputs.
DAC_TimerTrigger	Demonstrate how Timer triggers DAC to convert sine wave outputs.

Analog Comparator Controller (ACMP)

ACMP_ComapreDAC	Demonstrate how ACMP compares DAC output with ACMP1_N1 value.
ACMP_CompareVBG	Demonstrate analog comparator (ACMP) comparison by comparing ACMP1_P1 input and VBG voltage and shows the result on UART console.
ACMP_Wakeup	Use ACMP to wake up system from Power-down mode while comparator output changes.
ACMP_WindowCompare	Show how to monitor ACMP input with window compare function.
ACMP_WindowLatch	Demonstrate how to use ACMP window latch mode.

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.