M0A21 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.

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 M0A21 Series CMSIS BSP Driver Reference Guide.chm	This document describes the usage of drivers in M0A21 BSP.
NuMicro M0A21 Series CMSIS BSP Revision History.pdf	This document shows the revision history of M0A21 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

Hard_Fault_Sample	Show hard fault information when hard fault happened. The hard fault handler show 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. 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 M0A21 series MCU peripheral driver APIs.
Template	A project template for M0A21 series MCU.

4 SampleCode\ISP

ISP_RS485	In-System-Programming Sample code through RS485 interface.
ISP_UART	In-System-Programming Sample code through UART interface.
ISP_USCI_I2C	In-System-Programming Sample code through USCI I2C interface.

5 SampleCode\StdDriver

System Manager (SYS)

SYS_BODWakeup	Demonstrate how to wake up system from Power-down mode by brown-out detector interrupt.
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_IAP	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.
FMC_RW	Show FMC 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.

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).

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_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.

PWM Generator and Capture Timer (PWM)

PWM_240KHz_DutySwitch	Demonstrate how to set PWM0 channel 0 output 240 kHz waveform and switch duty in each 0.5%.
PWM_Capture	Capture the PWM Channel 2 waveform by PWM Channel 0.
PWM_DeadZone	Demonstrate how to use PWM Dead Time function.
PWM_DoubleBuffer_ PeriodLoadingMode	Change duty cycle and period of output waveform by PWM Double Buffer function.
PWM_DutySwitch	Change duty cycle of output waveform by configured period.
PWM_OutputWaveform	Demonstrate how to use PWM counter output waveform.
PWM_PDMA_Capture	Capture the PWM0 Channel 0 waveform by PWM0 Channel 2, and use PDMA to transfer captured data.
PWM_PDMA_Capture_ 1MHzSignal	Capture the PWM0 Channel 0 waveform by PWM0 Channel 2, and use PDMA to transfer captured data. The frequency of PWM Channel 0 is 1 MHz, which is used to test the maximum input frequency for PWM Capture function.
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_LIN	Demonstrate how to send data to LIN bus.
UART_LIN_Wakeup	Demonstrate how to wake up a device by LIN Wake-up Signal on 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.

Universal Serial Control Interface Controller - I²C Mode (USCI-I2C)

USCI_I2C_EEPROM	Show how to use USCI_I2C interface to access EEPROM.
USCI_I2C_Loopback	Show a Master how to access 7-bit address Slave (loopback).
USCI_I2C_Loopback_10bit	Show a Master how to access 10-bit address Slave (loopback).
USCI_I2C_Master	Show how a master accesses a slave. This sample code needs to work with USCI_I2C_Slave.
USCI_I2C_Master_10bit	Show how a master accesses a 10-bit address slave. This sample code need works with USCI_I2C_Slave_10bit.

USCI_I2C_MultiBytes_Master	Show how to set USCI_I2C Multi bytes API Read and Write data to Slave. This sample code needs to work with USCI_I2C_Slave.
USCI_I2C_SingleByte_Master	Show how to use USCI_I2C Single byte API Read and Write data to Slave. This sample code needs to work with USCI_I2C_Slave.
USCI_I2C_Slave	Show how a slave receives data from a master. This sample code needs to work with USCI_I2C_Master.
USCI_I2C_Slave_10bit	Show how a 10-bit address slave receives data from a master. This sample code need works with USCI_I2C_Master_10bit.
USCI_I2C_Wakeup_Slave	Show how to wake up USCI_I2C from Deep Sleep mode. This sample code needs to work with USCI_I2C_Master.

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

USCI_SPI_Loopback	Implement USCI_SPI0 Master loop back transfer. This sample code needs to connect USCI_SPI0_MISO pin and USCI_SPI0_MOSI pin together. It will compare the received data with transmitted data.
USCI_SPI_MasterMode	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.
USCI_SPI_PDMA_LoopTest	Demonstrate USCI_SPI data transfer with PDMA. USCI_SPI0 will be configured as Master mode and USCI_SPI1 will be configured as Slave mode. Both TX PDMA function and RX PDMA function will be enabled.
USCI_SPI_SlaveMode	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.

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

USCI_UART_AutoBaudRate	Show how to use auto baud rate detection function.
USCI_UART_Autoflow_Master	Transmit and receive data with auto flow control. This sample code needs to work with USCI_UART_Autoflow_Slave.
USCI_UART_Autoflow_Slave	Transmit and receive data with auto flow control. This sample code needs to work with USCI_UART_Autoflow_Master.
USCI_UART_PDMA	This is a USCI_UART PDMA demo and needs to connect USCI_UART TX and RX.
USCI_UART_RS485_Master	Transmit and receive data in RS485 mode. This sample code needs to work with USCI_UART_RS485_Slave.
USCI_UART_RS485_Slave	Transmit and receive data in RS485 mode. This sample code needs to work with USCI_UART_RS485_Master.
USCI_UART_TxRxFunction	Transmit and receive data from PC terminal via the RS232 interface.
USCI_UART_Wakeup	Show how to wake up system from Power-down mode by USCI interrupt in UART mode.

Controller Area Network Converter (CAN)

CAN_BasicMode_Rx	Demonstrate CAN bus receive a message with basic mode.
CAN_BasicMode_Tx	Demonstrate CAN bus transmit a message with basic mode.
CAN_NormalMode_RX	Demonstrate CAN bus receive a message with normal mode.
CAN_NormalMode_TX	Demonstrate CAN bus transmit a message with normal mode.

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.

Hardware Divider (HDIV)

HDIV	Demonstrate how to divide two signed integers by HDIV engine.

Analog-to-Digital Converter (ADC)

ADC_1411ksps_ContinuousScanMode	Demonstrate how to use HIRC as ADC clock source to achieve 1411 ksps ADC conversion rate.
ADC_ADINT_Trigger	Use ADINT interrupt to do the ADC Single-cycle scan conversion.
ADC_BandGap	Convert Band-gap (channel 29) and print conversion result.
ADC_BandGapCalculateAVDD	Demonstrate how to calculate battery voltage (AVdd) by using band-gap.
ADC_BurstMode	Perform A/D Conversion with ADC burst mode.
ADC_ContinuousScanMode	Perform A/D Conversion with ADC continuous scan mode.
ADC_PDMA_PWM_Trigger	Demonstrate how to trigger ADC by PWM and transfer conversion data by PDMA.
ADC_PWM_Trigger	Demonstrate how to trigger ADC by PWM.
ADC_ResultMonitor	Monitor the conversion result of channel 2 by the digital compare function.

ADC_SingleCycleScanMode	Perform A/D Conversion with ADC single cycle scan mode.
ADC_SingleMode	Perform A/D Conversion with ADC single mode.
ADC_STADC_Trigger	Show how to trigger ADC by STADC pin.
ADC_SwTrg_Trigger	Trigger ADC by writing ADC software trigger register.
ADC_TempSensor	Convert temperature sensor (channel 30) and print conversion result.
ADC_Timer_Trigger	Show how to trigger ADC by Timer.

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 compare DAC output with ACMP1_N1 value.
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.