NUC1262 Series CMSIS BSP Guide

Directory Introduction for 32-bit NuMicro™ Family

Directory Information

Please extract the “NUC1262Series_BSP_CMSIS_V3.00.004.zip” file firstly, and then put the “NUC1262Series_BSP_CMSIS_V3.00.004” folder into the working folder (e.g. .\Nuvoton\BSP Library\).

This BSP folder contents:

Document	Device driver reference manual and reversion history.
Library	Device driver header and source files.
SampleCode	Device 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 4

2 LIBRARY 5

3 SAMPLE CODE 6

4 SAMPLECODE\ISP 7

5 SAMPLECODE\REGBASED 8

Clock Controller (CLK) 8
Flash Memory Controller (FMC) 8
General Purpose I/O (GPIO) 8
PDMA Controller (PDMA) 9
Timer Controller (TIMER) 9
Watchdog Timer (WDT) 9
Window Watchdog Timer (WWDT) 10
Basic PWM Generator and Capture Timer (BPWM) 10
UART Interface Controller (UART) 10
I²S Controller (I²S) 10
Serial Peripheral Interface (SPI) 11
I²C Serial Interface Controller (I²C) 11
CRC Controller (CRC) 12
Analog-to-Digital Converter (ADC) 13
LED Light Strip Interface (LLSI) 13

6 SAMPLECODE\STDDRIVER 14

System Manager (SYS) 14
Clock Controller (CLK) 14
Flash Memory Controller (FMC) 14
General Purpose I/O (GPIO) 14
Timer Controller (TIMER) 15
Watchdog Timer (WDT) 15
Window Watchdog Timer (WWDT) 15
Basic PWM Generator and Capture Timer (BPWM) 15
UART Interface Controller (UART) 16

I²S Controller (I²S) 16
Serial Peripheral Interface (SPI) 17
I²C Serial Interface Controller (I²C) 17
CRC Controller (CRC) 18
Analog-to-Digital Converter (ADC) 19
LED Light Strip Interface (LLSI) 19
PDMA Controller (PDMA) 19
USB Device Controller (USBD) 20

1 Document

CMSIS.html	Introduction of CMSIS version 5. CMSIS components included CMSIS-CORE, CMSIS-Driver, CMSIS-DSP, etc. ● CMSIS-CORE: API for the Cortex-M0 processor core and peripherals. ● CMSIS-Driver: Defines generic peripheral driver interfaces for middleware making it reusable across supported devices. ● CMSIS-DSP: DSP Library Collection with over 60 Functions for various data types: fix-point (fractional q7, q15, q31) and single precision floating-point (32-bit).
NuMicro NUC1262 Series CMSIS BSP Revision History.pdf	The revision history of NUC1262Series BSP.
NuMicro NUC1262 Series Driver Reference Guide.chm	The usage of drivers in NUC1262 Series BSP.

2 Library

CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) V5.0 definitions by ARM® Corp.
Device	CMSIS compliant device header file.
StdDriver	All peripheral driver header and source files.

3 Sample Code

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 occur. 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.
RegBased	The sample codes which access control registers directly.
Semihost	Show how to print and get character through IDE console window.
StdDriver	Demonstrate the usage of NUC1262 series MCU peripheral driver APIs.
Template	A project template for NUC1262 series MCU.

4 SampleCode\ISP

ISP_DFU	In-System-Programming Sample code through USB interface and following Device Firmware Upgrade Class Specification.
ISP_HID	In-System-Programming Sample code through USB HID interface.
ISP_I2C	In-System-Programming Sample code through I2C interface.
ISP_RS485	In-System-Programming Sample code through RS485 interface.
ISP_SPI	In-System-Programming Sample code through SPI interface.
ISP_UART	In-System-Programming Sample code through UART interface.

5 SampleCode\RegBased

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

Clock Controller (CLK)

CLK_ClockDetector	Show the usage of clock fail detector and clock frequency monitor function.

Flash Memory Controller (FMC)

FMC_ExeInSRAM	Implement a code and execute the code in SRAM to program embedded Flash (support KEIL MDK only).
FMC_IAP	Show how to call LDROM functions from APROM. The code in APROM will look up the table at 0x100E00 to get the address of function of LDROM and call the function.
FMC_MultiBoot	Implement a multi-boot system to boot from different applications in APROM. A LDROM code and four APROM code are implemented in this sample code.
FMC_RW	Demonstrate how to read/program embedded Flash by ISP function.
FMC_SPROM	Demonstrate how to set SPROM in security mode.

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/output control.
GPIO_PowerDown	Show how to wake up system from Power-down mode by GPIO interrupt.

PDMA Controller (PDMA)

PDMA	Use PDMA Channel 2 to transfer data from memory to memory.
PDMA_ScatterGather	Use PDMA0 channel 4 to transfer data from memory to memory by scatter-gather mode.
PDMA_ScatterGather_PingPongBuffer	Use PDMA0 to implement Ping-Pong buffer by scatter-gather mode(memory to memory).

Timer Controller (TIMER)

TIMER_CaptureCounter	Show how to use the timer2 capture function to capture timer2 counter value.
TIMER_EventCounter	Show how to use the timer2 capture function to capture timer2 counter value.
TIMER_PeriodicINT	Implement timer counting in periodic mode.
TIMER_TimeoutWakeup	Use timer0 periodic time-out interrupt event to wake up system.

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_CompareINT	Show how to reload the WWDT counter value.

Basic PWM Generator and Capture Timer (BPWM)

BPWM_Capture	Use BPWM0 channel 0 to capture the BPWM1 channel 0 waveform.
BPWM_DoubleBuffer_PeriodLoadingMode	Change duty cycle and period of output waveform by BPWM Double Buffer function(Period loading mode).
BPWM_OutputWaveform	Demonstrate how to use BPWM counter output waveform.
BPWM_SwitchDuty	Change duty cycle of output waveform by configured period.
BPWM_SyncStart	Demonstrate how to use BPWM 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 data in UART IrDA mode.
UART_PDMA	Transmit and receive UART data with PDMA.
UART_RS485	Transmit and receive data in UART RS485 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.

I²S Controller (I²S)

I2S_Master	Configure SPI1 as I2S Master mode and demonstrate how I2S works in Master mode. This sample code needs to work with I2S_Slave.

I2S_PDMA_NAU8822	This is an I2S demo with PDMA function connected with NAU8822 codec.
I2S_PDMA_Play	This is an I2S demo for playing data and demonstrating how I2S works with PDMA.
I2S_PDMA_PlayRecord	This is an I2S demo for playing and recording data with PDMA function.
I2S_PDMA_Record	This is an I2S demo for recording data and demonstrating how I2S works with PDMA.
I2S_Slave	Configure SPI1 as I2S Slave mode and demonstrate how I2S works in Slave mode. This sample code needs to work with I2S_Master.

Serial Peripheral Interface (SPI)

SPI_Loopback	Implement SPI Master loop back transfer. This sample code needs to connect MISO pin and MOSI pin together. It will compare the received data with transmitted data.
SPI_MasterFIFOMode	Configure SPI0 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 sample code.
SPI_PDMA_LoopTest	Demonstrate SPI data transfer with PDMA. QSPI0 will be configured as Master mode and SPI1 will be configured as Slave mode. Both TX PDMA function and RX PDMA function will be enabled.
SPI_SlaveFIFOMode	Configure SPI0 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 sample code.

I²C Serial Interface Controller (I²C)

I2C_EEPROM	Demonstrate how to access EEPROM through a I²C interface
I2C_GCMode_Master	Demonstrate how a Master uses I²C address 0x0 to write data to I²C Slave. This sample code needs to work with

	I2C_GCMode_Slave sample code.
I2C_GCMode_Slave	Demonstrate how to receive Master data in GC (General Call) mode. This sample code needs to work with I2C_GCMode_Master sample code.
I2C_Loopback	Demonstrate how a Master accesses Slave.
I2C_Loopback_10bit	Demonstrate how a Master uses 10-bit addressing to access a Slave.
I2C_Master	Demonstrate how a Master accesses Slave. This sample code needs to work with I2C_Slave sample code.
I2C_Master_PDMA	Demonstrate how a Master accesses Slave using PDMA TX mode and PDMA RX mode.
I2C_Slave	Demonstrate how to set I^2C in slave mode to receive the data from a Master. This sample code needs to work with I2C_Master sample code.
I2C_Slave_PDMA	Demonstrate how a Slave uses PDMA Rx mode receive data from a Master.
I2C_SMBUS	Demonstrate how I2C SMBUS works.
I2C_Wakeup_Slave	Demonstrate how to set I^2C to wake up MCU from Power-down mode. This sample code needs to work with I2C_Master sample code.
I2C_Wakeup_Slave_10bit	Demonstrate how to set I2C uses 10-bit addressing to wake up MCU from Power-down mode. This sample code needs to work with I2C_Master.

CRC Controller (CRC)

CRC_CCITT	Implement CRC in CRC-CCITT mode and get the CRC checksum result.
CRC_CRC8	Implement CRC in CRC-8 mode and get the CRC checksum result.
CRC_CRC32	Implement CRC in CRC-32 mode with PDMA transfer.

Analog-to-Digital Converter (ADC)

ADC_BandGap	Convert Band-gap (channel 29) and print conversion result.
ADC_ContinuousScanMode	Perform A/D Conversion with ADC continuous scan mode.
ADC_PDMA_SingleCycleScanMode	Perform A/D Conversion with ADC single cycle scan mode and transfer result by PDMA.
ADC_PwmTrigger	Demonstrate how to trigger ADC by PWM.
ADC_ResultMonitor	Demonstrate how to use the digital compare function to monitor the conversion result of channel 2.
ADC_SingleCycleScanMode	Demonstrate how to use single cycle scan mode and finishes one cycle of conversion for the specified channels.
ADC_SingleMode	Demonstrate how to use single mode and finishes the conversion of the specified channel.
ADC_TempSensor	Convert temperature sensor (channel 30) and print conversion result.

LED Light Strip Interface (LLSI)

LLSI_Marquee	This is a LLSI demo for marquee display in software mode. It needs to be used with WS2812 LED strip.
LLSI_PDMA_Marquee	This is a LLSI demo for marquee display in PDMA mode. It needs to be used with WS2812 LED strip.

6 SampleCode\StdDriver

System Manager (SYS)

SYS_BODWakeup	Show how to wake up system form Power-down mode by brown-out detector interrupt.
SYS_PLLClockOutput	Demonstrate how to change system clock to different PLL frequency and output system clock from 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.
SYS_VoltageDetector	Show how to use voltage detector to detect pin input voltage.

Clock Controller (CLK)

CLK_ClockDetector	Show the usage of clock fail detector and clock frequency monitor function.

Flash Memory Controller (FMC)

FMC_ExeInSRAM	Implement a code and execute in SRAM to program embedded Flash (support KEIL MDK only).
FMC_IAP	Show how to set VECMAP to LDROM and reboot to LDROM from APROM.
FMC_RW	Demonstrate how to read/program embedded Flash by ISP function.
FMC_SPROM	Demonstrate how to set SPROM in security mode.

General Purpose I/O (GPIO)

GPIO_EINTAndDebounce	Demonstrate how to use GPIO external interrupt function and de-bounce function.
GPIO_INT	Demonstrate how to use GPIO interrupt function.

GPIO_OutputInput	Demonstrate how to set GPIO pin mode and use pin data input/output control.
GPIO_PowerDown	Demonstrate how to wake-up form Power-down mode by GPIO interrupt.

Timer Controller (TIMER)

TIMER_CaptureCounter	Show how to use the timer2 capture function to capture timer2 counter value.
TIMER_Delay	Show how to use timer0 to create various delay time.
TIMER_EventCounter	Implement timer1 event counter function to count the external input event.
TIMER_InterTimerTriggerMode	Demonstrate how to use Inter-Timer trigger function.
TIMER_PeriodicINT	Implement timer counting in periodic mode.
TIMER_TimeoutWakeup	Use timer0 periodic time-out interrupt event to wake up system.
TIMER_ToggleOut	Implement timer counting in toggle-output mode.

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_CompareINT	Select one WWDT window compare value to generate window compare match interrupt event.

Basic PWM Generator and Capture Timer (BPWM)

BPWM_Capture	Use BPWM0 channel 0 to capture the BPWM1 channel 0

	waveform.
BPWM_DoubleBuffer_PeriodLoadingMode	Change duty cycle and period of output waveform by BPWM Double Buffer function.(Period loading mode)
BPWM_DutySwitch	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.

UART Interface Controller (UART)

UART_AutoBaudRate	Show how to use auto baud rate detection function.
UART_Autoflow	Transmit and receive data with auto flow control.
UART_IrDA	Transmit and receive data in UART IrDA mode.
UART_PDMA	Transmit and receive UART data with PDMA.
UART_RS485	Transmit and receive data in UART RS485 mode.
UART_TxRx_Function	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.

I²S Controller (I²S)

I2S_Master	Configure SPI1 as I2S Master mode and demonstrate how I2S works in Master mode. This sample code needs to work with I2S_Slave.
I2S_PDMA_NAU8822	This is an I2S demo with PDMA function connected with NAU8822 codec.
I2S_PDMA_Play	This is an I2S demo for playing data and demonstrating how I2S works with PDMA.
I2S_PDMA_PlayRecord	This is an I2S demo for playing and recording data with PDMA function.

I2S_PDMA_Record	This is an I2S demo for recording data and demonstrating how I2S works with PDMA.
I2S_Slave	Configure SPI1 as I2S Slave mode and demonstrate how I2S works in Slave mode. This sample code needs to work with I2S_Master.

Serial Peripheral Interface (SPI)

SPI_Loopback	Implement SPI Master loop back transfer. This sample code needs to connect SPI0_MISO0 pin and SPI0_MOSI0 pin together. It will compare the received data with transmitted data.
SPI_MasterFIFOMode	Configure SPI0 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 sample code.
SPI_PDMA_LoopTest	Demonstrate SPI data transfer with PDMA. SPI0 will be configured as Master mode and SPI1 will be configured as Slave mode. Both TX PDMA function and RX PDMA function will be enabled.
SPI_SlaveFIFOMode	Configure SPI0 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 sample code.

I²C Serial Interface Controller (I²C)

I2C_EEPROM	Demonstrate how to access EEPROM by I²C interface.
I2C_GCMode_Master	Demonstrate how a Master uses I²C address 0x0 to write data to I²C Slave. Needs to work with I2C_GCMode_SLAVE sample code.
I2C_GCMode_Slave	Demonstrate how to receive Master data in GC (General Call) mode. Needs to work with I2C_GCMode_MASTER sample code.
I2C_Loopback	Demonstrate how a Master access Slave.

I2C_Loopback_10bit	Demonstrate how a Master uses 10-bit addressing to access a Slave.
I2C_Master	Demonstrate how a Master access Slave. Needs to work with I2C_SLAVE sample code.
I2C_Slave	Demonstrate how to set I^2C in slave mode to receive the data of a Master. Needs to work with I2C_MASTER sample code.
I2C_Master_PDMA	Demonstrate how a Master accesses Slave using PDMA TX mode and PDMA RX mode.
I2C_Slave_PDMA	Demonstrate how a Slave uses PDMA Rx mode receive data from a Master.
I2C_MultiBytes_Master	Show how to set I2C use Multi bytes API Read and Write data to Slave. Needs to work with I2C_SLAVE sample code.
I2C_SingleByte_Master	Show how to use I2C Signle byte API Read and Write data to Slave. Needs to work with I2C_SLAVE sample code
I2C_SMBUS	Demonstrate how I2C SMBUS works.
I2C_Wakeup_Slave	Demonstrate how to set I2C to wake up MCU from Power-down mode. This sample code needs to work with I2C_MASTER.
I2C_Wakeup_Slave_10bit	Demonstrate how to set I2C uses 10-bit addressing to wake up MCU from Power-down mode. This sample code needs to work withI2C_MASTER.

CRC Controller (CRC)

CRC_CCITT	Implement CRC in CRC-CCITT mode and get the CRC checksum result.
CRC_CRC32	Implement CRC in CRC-32 mode with PDMA transfer.
CRC_CRC8	Implement CRC in CRC-8 mode and get the CRC checksum result.

Analog-to-Digital Converter (ADC)

ADC_BandGap	Convert Band-gap (channel 29) and print conversion result.
ADC_ContinuousScanMode	Perform A/D Conversion with ADC continuous scan mode.
ADC_PDMA_SingleCycleScanMode	Perform A/D Conversion with ADC single cycle scan mode and transfer result by PDMA.
ADC_PwmTrigger	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_TempSensor	Convert temperature sensor (channel 30) and print conversion result.

LED Light Strip Interface (LLSI)

LLSI_Marquee	This is a LLSI demo for marquee display in software mode. It needs to be used with WS2812 LED strip.
LLSI_PDMA_Marquee	This is a LLSI demo for marquee display in PDMA mode. It needs to be used with WS2812 LED strip.

PDMA Controller (PDMA)

PDMA	Use PDMA Channel 2 to transfer data from memory to memory.
PDMA_ScatterGather	Use PDMA0 channel 4 to transfer data from memory to memory by scatter-gather mode.
PDMA_ScatterGather_PingPongBuffer	Use PDMA0 to implement Ping-Pong buffer by scatter-gather mode(memory to memory).

USB Device Controller (USBD)

USBD_Audio_NAU8822	This sample code demonstrates how to implement a USB audio class device. NAU8822 is used in this sample code to play the audio data from Host. It also supports to record data from NAU8822 to Host.
USBD_HID_Keyboard	Show how to implement a USB keyboard device. This sample code supports to use GPIO to simulate key input.
USBD_HID_Mouse	Show how to implement a USB mouse device. The mouse cursor will move automatically when this mouse device connecting to PC by USB.
USBD_HID_Mouse2	Demonstrate how to implement a USB mouse device. It uses PC0 ~ PC5 to control mouse directions and mouse keys. It also supports USB suspend and remote wakeup.
USBD_HID_MouseKeyboard	Demonstrate how to implement a USB mouse function and a USB keyboard on the same USB device. The mouse cursor will move automatically when this mouse device connecting to PC. This sample code uses a GPIO to simulate key input.
USBD_HID_Transfer	Transfer data between USB device and PC through USB HID interface. A windows tool is also included in this sample code to connect with USB device.
USBD_HID_Transfer_And_Keyboard	Demonstrate how to implement a composite device. (HID Transfer and keyboard) Transfer data between USB device and PC through USB HID interface. A windows tool is also included in this sample code to connect with USB device.
USBD_HID_Transfer_And_MSC	Demonstrate how to implement a composite device. (HID Transfer and Mass storage) Transfer data between USB device and PC through USB HID interface. A windows tool is also included in this sample code to connect with a USB device.
USBD_MassStorage_CDROM	Demonstrate how to simulate a USB CD-ROM device.
USBD_MassStorage_DataFlash	Demonstrate how to implement a USB mass storage. It uses embedded Data Flash as storage.
USBD_MassStorage_SDCard	Use SD card as storage to implement a USB mass storage device.

USBD_Micro_Printer	Show how to implement a USB micro printer device.
USBD_Printer_And_HID_Transfer	Demonstrate how to implement a composite device (USB micro printer device and HID Transfer). Transfer data between a USB device and PC through a USB HID interface. A windows tool is also included in this sample code to connect with a USB device.
USBD_VCOM_And_HID_Keyboard	Implement a USB composite device with virtual COM port and keyboard functions.
USBD_VCOM_And_HID_Transfer	Demonstrate how to implement a composite device (VCOM and HID Transfer). It supports one virtual COM port and transfers data between a USB device and PC through a USB HID interface. A windows tool is also included in this sample code to connect with a USB device.
USBD_VCOM_and_MassStorage	Implement a USB composite device. It supports one virtual COM port and one USB mass storage device.
USBD_VCOM_DualPort	Demonstrate how to implement a USB dual virtual COM port device.
USBD_VCOM_SinglePort	Implement a USB virtual COM port device. It supports one virtual COM port.

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.