NUC1261 Series CMSIS BSP Guide

Directory Introduction for 32-bit NuMicro™ Family

Directory Information

Please extract the “NUC1261Series_BSP_CMSIS_V3.00.005.zip” file firstly, and then put the “NUC1261Series_BSP_CMSIS_V3.00.005” 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.

TABLE OF CONTENTS

1 Document

CMSIS.html	Introduction of CMSIS version 4.5.0. 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 NUC1261 Series CMSIS BSP Revision History.pdf	The revision history of NUC1261 Series BSP.
NuMicro NUC1261 Series CMSIS BSP Driver Reference Guide.chm	The usage of drivers in NUC1261 Series BSP.

2 Library

CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) V4.5.0 definitions by ARM® Corp.
Device	CMSIS compliant device header file.
SmartcardLib	Library for accessing a smartcard.
StdDriver	All peripheral driver header and source files.

3 Sample Code

CardReader	CCID (Circuit card interface device) sample code for smart card interface.
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.
Semihost	Show how to print and get character through IDE console window.
RegBased	The sample codes which access control registers directly.
StdDriver	Demonstrate the usage of NUC1261 series MCU peripheral driver APIs.
Template	A project template for NUC1261 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

System Manager (SYS)

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_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_PingPongBuffer	Use PDMA to implement Ping-Pong buffer by scatter-gather mode (memory to memory).
PDMA_Scatter_Gather	Use PDMA Channel 4 to transfer data from memory to memory by scatter-gather mode.

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_PWM_ChangeDuty	Change duty cycle and period of output waveform in PWM down count type.
TIMER_PWM_DeadTime	Demonstrate Timer PWM Complementary mode and Dead-Time function.
TIMER_PWM_OuputWaveform	Demonstrate output different duty waveform in Timer0~Timer3 PWM.
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.

Real Timer Clock (RTC)

RTC_AlarmWakeup	Use RTC alarm interrupt event to wake up system.
RTC_TimeAndTick	Show the current RTC data/time per tick.

PWM Generator and Capture Timer (PWM)

PWM_Capture	Capture the PWM1 Channel 0 waveform by PWM1 Channel 2.
PWM_DeadZone	Demonstrate how to use PWM Dead Zone function.
PWM_DoubleBuffer_PeriodLoadingMode	Change duty cycle and period of output waveform by PWM Double Buffer function (Period loading mode).
PWM_DutySwitch	Change duty cycle of output waveform by configured period.
PWM_OutputWaveform	Demonstrate how to use PWM output waveform.
PWM_PDMA_Capture	Capture the PWM1 Channel 0 waveform by PWM1 Channel 2, and use PDMA to transfer

	captured data.
PWM_SyncStart	Demonstrate how to use PWM counter synchronous start function.

UART Interface Controller (UART)

UART_AutoBaudRate_Master	Show how to use auto baud rate detection function. This sample code needs to work with UART_AutoBaudRate_Slave.
UART_AutoBaudRate_Slave	Show how to use auto baud rate detection function. This sample code needs to work with UART_AutoBaudRate_Master.
UART_Autoflow_Master	Transmit and receive data with auto flow control. This sample code needs to work with UART_Autoflow_Slave.
UART_Autoflow_Slave	Transmit and receive data with auto flow control. This sample code needs to work with UART_Autoflow_Master.
UART_IrDA_Master	Transmit and receive data in UART IrDA mode. This sample code needs to work with UART_IrDA_Slave.
UART_IrDA_Slave	Transmit and receive data in UART IrDA mode. This sample code needs to work with UART_IrDA_Master.
UART_LIN	Transmit LIN frame including header and response in UART LIN mode.
UART_PDMA	Transmit and receive UART data with PDMA.
UART_RS485_Master	Transmit and receive data in UART RS485 mode. This sample code needs to work with UART_RS485_Slave.
UART_RS485_Slave	Transmit and receive data in UART RS485 mode. This sample code needs to work with UART_RS485_Master.
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.

Smartcard Host Interface (SC)

SCUART_TxRx	Show Smartcard UART by connecting PA.0 and PA.1 pins.
SC_ReadATR	Read Smartcard ATR from the SC0 port.

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.
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.

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

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.
I2C_GCMode_Slave	Demonstrate how to receive Master data in GC (General Call) mode. This sample code needs to work with I2C_GCMode_Master.
I2C_Loopback	Demonstrate how a Master accesses a Slave.
I2C_Master	Demonstrate how a Master accesses a Slave. This sample code needs to work with I2C_Slave.
I2C_Master_PDMA	Demonstrate how a Master accesses Slave using PDMA TX mode and PDMA RX mode.
I2C_Slave	Demonstrate how to set I2C in slave mode to receive data from a Master. This sample code needs to work with I2C_Master.
I2C_Slave_PDMA	Demonstrate how a Slave uses PDMA Rx mode to receive data from a Master.
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.

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

USCI_UART_AutoBaudRate_Master	Show how to use auto baud rate detection function. This sample code needs to work with USCI_UART_AutoBaudRate_Slave.
USCI_UART_AutoBaudRate_Slave	Show how to use auto baud rate detection function. This sample code needs to work with USCI_UART_AutoBaudRate_Master.
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_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 through RS232 interface.
USCI_UART_Wakeup	Show how to wake up system from Power-down mode by USCI interrupt in UART mode.

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

USCI_I2C_EEPROM	Demonstrate how to access EEPROM through a USCI_I2C interface
USCI_I2C_Loopback	Demonstrate how a Master accesses Slave.
USCI_I2C_Loopback_10bit	Demonstrate how a Master uses10-bit addressing access Slave.
USCI_I2C_Master	Demonstrate how a Master accesses Slave. This sample code needs to work with I2C_Slave.
USCI_I2C_Master_10bit	Demonstrate how a Master uses 10-bit addressing access Slave. This sample code needs to work with I2C_Slave.
USCI_I2C_Slave	Demonstrate how to set I2C in Slave mode to receive data from a Master. This sample code needs to work with I2C_Master.
USCI_I2C_Slave_10bit	Demonstrate how to set I2C in 10-bit addressing slave mode to receive data from a Master. This sample code needs to work with I2C_Master.
USCI_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.

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

USCI_SPI_Loopback	Implement USCI_SPI1 Master loop back transfer. This sample code needs to connect USCI_SPI1_MISO pin and USCI_SPI1_MOSI pin together. It will compare the received data with transmitted data.
USCI_SPI_MasterMode	Configure USCI_SPI1 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_SlaveMode	Configure USCI_SPI1 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.

I^{2}S Controller (I^{2}S)

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.

External Bus Interface (EBI)

EBI_NOR	Configure EBI interface to access MX29LV320T (NOR Flash) on EBI interface.
EBI_SRAM	Configure EBI interface to access BS616LV4017 (SRAM) on EBI interface.

CRC Controller (CRC)

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

Analog-to-Digital Converter (ADC)

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.

Analog Comparator Controller (ACMP)

ACMP	Demonstrate how Analog Comparator (ACMP) works with internal band-gap voltage.

ACMP_Wakeup	Show how to wake up MCU from Power-down mode by ACMP wake-up function.

Hardware Divider (HDIV)

HDIV	Show how to use divider API and how to use hardware divider by control registers.

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	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_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	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_PingPongBuffer	Use PDMA to implement Ping-Pong buffer by scatter-gather mode (memory to memory).
PDMA_ScatterGather	Use PDMA Channel 4 to transfer data from memory to memory by scatter-gather mode.

Timer Controller (TIMER)

TIMER_ACMPTrigger	Show how to use ACMP0 to trigger Timer capture event.
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_PWM_Brake	Generate Timer brake event by Timer brake pin.
TIMER_PWM_ChangeDuty	Change duty cycle and period of output waveform in PWM down count type.

TIMER_PWM_DeadTime	Demonstrate Timer PWM Complementary mode and Dead-Time function.
TIMER_PWM_OutputWaveform	Demonstrate output different duty waveform in Timer0~Timer3 PWM.
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	Show how to reload the WWDT counter value.

Real Timer Clock (RTC)

RTC_AlarmWakeup	Use RTC alarm interrupt event to wake up system.
RTC_TimeAndTick	Show the current RTC data/time per tick.

PWM Generator and Capture Timer (PWM)

PWM_Capture	Capture the PWM1 Channel 0 waveform by PWM1 Channel 2.
PWM_DeadZone	Demonstrate how to use PWM Dead Zone function.
PWM_DoubleBuffer_PeriodLoadingMode	Change duty cycle and period of output waveform by PWM Double Buffer function

	(Period loading mode).
PWM_DutySwitch	Change duty cycle of output waveform by configured period.
PWM_OutputWaveform	Demonstrate how to use PWM output waveform.
PWM_PDMA_Capture	Capture the PWM1 Channel 0 waveform by PWM1 Channel 2, and use PDMA to transfer captured data.
PWM_SyncStart	Demonstrate how to use PWM counter synchronous start function.

UART Interface Controller (UART)

UART_AutoBaudRate_Master	Show how to use auto baud rate detection function. This sample code needs to work with UART_AutoBaudRate_Slave.
UART_AutoBaudRate_Slave	Show how to use auto baud rate detection function. This sample code needs to work with UART_AutoBaudRate_Master.
UART_Autoflow_Master	Transmit and receive data with auto flow control. This sample code needs to work with UART_Autoflow_Slave.
UART_Autoflow_Slave	Transmit and receive data with auto flow control. This sample code needs to work with UART_Autoflow_Master.
UART_IrDA_Master	Transmit and receive data in UART IrDA mode. This sample code needs to work with UART_IrDA_Slave.
UART_IrDA_Slave	Transmit and receive data in UART IrDA mode. This sample code needs to work with UART_IrDA_Master.
UART_LIN	Transmit LIN frame including header and response in UART LIN mode.
UART_PDMA	Transmit and receive UART data with PDMA.

UART_RS485_Master	Transmit and receive data in UART RS485 mode. This sample code needs to work with UART_RS485_Slave.
UART_RS485_Slave	Transmit and receive data in UART RS485 mode. This sample code needs to work with UART_RS485_Master.
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.

Smart Card Host Interface (SC)

SCUART_TxRx	Show Smartcard UART by connecting PA.0 and PA.1 pins.
SC_ReadATR	Read Smartcard ATR from the SC0 port.
SC_ReadSimPhoneBook	Read SIM phone book from the SC0 port.

I^{2}S Controller (I^{2}S)

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.

USB Device Controller (USBD)

USBD_Audio_NAU8822	Demonstrate 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 wake-up.
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	Use embedded Data Flash 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.

Serial Peripheral Interface (SPI)

SPI_Loopback	Implement SPI Master loop back transfer. This sample code needs to connect MISO_0 pin and MOSI_0 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.
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.

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

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.
I2C_GCMode_Slave	Demonstrate how to receive Master data in GC (General Call) mode. This sample code needs to work with I2C_GCMode_Master.
I2C_Loopback	Demonstrate how a Master accesses Slave.
I2C_Master	Demonstrate how a Master accesses Slave. This sample code needs to work with I2C_Slave.
I2C_Master_PDMA	Demonstrate how a Master accesses Slave using PDMA TX mode and PDMA RX mode.
I2C_Slave	Demonstrate how to set I2C in slave mode to receive the data from a Master. This sample code needs to work with I2C_Master.
I2C_Slave_PDMA	Demonstrate how a Slave uses PDMA Rx mode

	receive data from a Master.
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.

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

USCI_UART_AutoBaudRate_Master	Show how to use auto baud rate detection function. This sample code needs to work with USCI_UART_AutoBaudRate_Slave.
USCI_UART_AutoBaudRate_Slave	Show how to use auto baud rate detection function. This sample code needs to work with USCI_UART_AutoBaudRate_Master.
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_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 through a RS232 interface.
USCI_UART_Wakeup	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)

USCI_SPI_Loopback	Implement USCI_SPI1 Master loop back transfer. This sample code needs to connect USCI_SPI1_MISO pin and USCI_SPI1_MOSI pin together. It will compare the received data

	with transmitted data.
USCI_SPI_MasterMode	Configure USCI_SPI1 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_SlaveMode	Configure USCI_SPI1 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 - I^2C Mode (USCI-I2C)

USCI_I2C_EEPROM	Demonstrate how to access EEPROM through a USCI_I2C interface.
USCI_I2C_Loopback	Demonstrate how a Master accesses Slave.
USCI_I2C_Loopback_10bit	Demonstrate how a Master uses 10-bit addressing access Slave.
USCI_I2C_Master	Demonstrate how a Master accesses Slave. This sample code needs to work with I2C_Slave.
USCI_I2C_Master_10bit	Demonstrate how a Master use 10-bit addressing access Slave. This sample code needs to work with I2C_Slave.
USCI_I2C_Slave	Demonstrate how to set I2C in slave mode to receive the data from a Master. This sample code needs to work with I2C_Master.
USCI_I2C_Slave_10bit	Demonstrate how to set I2C in 10-bit addressing slave mode to receive the data from a Master. This sample code needs to work with I2C_Master.
USCI_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.

External Bus Interface (EBI)

EBI_NOR	Configure EBI interface to access MX29LV320T (NOR Flash) on EBI interface.
EBI_SRAM	Configure EBI interface to access BS616LV4017 (SRAM) on EBI interface.

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_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.

Analog Comparator Controller (ACMP)

ACMP	Demonstrate how Analog Comparator (ACMP) works with internal band-gap voltage.

ACMP_Wakeup	Show how to wake up MCU from Power-down mode by ACMP wake-up function.

Hardware Divider (HDIV)

HDIV	Show how to use divider API and how to use hardware divider by control registers.

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.