M261 Series CMSIS BSP Guide

Directory Introduction for 32-bit NuMicro™ Family

Directory Information

Please extract the “M261Series_BSP_CMSIS_V3.00.002.zip” file firstly, and then put the “M261Series_BSP_CMSIS_V3.00.002” 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.
ThirdParty	Libraries of third parties.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1 .\Document\

CMSIS.html	Introduction of CMSIS version 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 M261 Series CMSIS BSP Revision History.pdf	The revision history of M261 Series BSP.
NuMicro M261 Series Driver Reference Guide.chm	The usage of drivers in M261 Series BSP.

2 .\Library\

CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) V5.0 definitions by ARM® Corp.
Device	CMSIS compliant device header file.
NuMaker	Specific libraries for M261 NuMaker board.
SmartcardLib	Library for accessing a smartcard.
StdDriver	All peripheral driver header and source files.
UsbHostLib	USB host library source code.

3 .\Sample Code\

AttackDetection	Sample codes for non-invasive physical attack detection.
CardReader	USB CCID Smartcard Reader 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.
MKROM	Demonstrate the usage of M261 MKROM libraries, and show how to generate a secure boot image for Secure Boot Verification.
PowerManagement	Power management sample code.
Semihost	Show how to print and get character through IDE console window.
StdDriver	Demonstrate the usage of M261 series MCU peripheral driver APIs.
XOM	Demonstrate how to create XOM library and use it.

4 .\ThirdParty\

emWin	Segger embedded graphic library – emWin.
FatFs	An open source FAT/exFAT file system library.

5 .\SampleCode\AttackDetection

CLK_FrequencyDetection	Show the usage of clock fail detector and clock frequency monitor function.
EADC_TemperatureDetection	Show how to measure temperature by EADC.
EADC_VoltageDetection	Show how to measure AVDD voltage by EADC.
RTC_FrequencyDetection	Show the usage of LXT clock frequency monitor function.
RTC_TamperDetection	Show the usage of RTC static and dynamic tamper function.
SYS_ResetDetection	Demonstrate the methods of detecting reset abnormalities.

6 .\SampleCode\ISP

ISP_CAN	In-System-Programming Sample code through CAN interface.
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.

7 .\SampleCode\MKROM

CRYPTOLibDemo	Demonstrate the usage of M261 MKROM Crypto libraries on AES, ECC, SHA, TDES and TRNG applications.
SecureBootDemo	Demonstrate how to create a secure boot system, including generates a secure boot image (NuBL2), secure code (NuBL32) and non-secure code (NuBL33). Therefore, the Secure Bootloader (NuBL1) can authenticate NuBL2, and NuBL2 can authenticate NuBL32 and NuBL33.

8 .\SampleCode\NuMaker

Blinky	A simple LED toggle sample code. It could be used as the startup of M261 NuMaker board.
DebugUART	A simple debug message print demo.

9 .\SampleCode\PowerManagement

SYS_DPDMode_Wakeup	Show how to wake up system form DPD Power-down mode by Wake-up pin(PC.0) or Wake-up Timer or RTC Tick or RTC Alarm or RTC Tamper 0.
SYS_PowerDownMode	Show how to enter to different Power-down mode and wake-up by RTC.
SYS_PowerMode	Show how to set different core voltage and main voltage regulator type.
SYS_SPDMode_Wakeup	Show how to wake up system form SPD Power-down mode by Wake-up pin(PC.0) or Wake-up Timer or Wake-up ACMP or RTC Tick or RTC Alarm and RTC Tamper 0.
SYS_SPDMode_WakeupAndReturn	Show how to continue executing code after wake-up form SPD Power-down mode by SRAM data retention function.
SYS_SRAMPowerMode	Show how to select SRAM power mode in system Power-down mode.

10 .\SampleCode\StdDriver

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.
BPWM_Capture	Use BPWM0 Channel 0 to capture the BPWM1 Channel 0 Waveform.
BPWM_DoubleBuffer	Change duty cycle and period of output waveform by BPWM Double Buffer function.
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.
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_BasicMode_Tx_Rx	Demonstrate CAN bus transmit and receive a message with basic mode by connecting CAN0 and CAN1 to the same CAN bus.
CAN_NormalMode_Rx	Demonstrate CAN bus receive a message with normal mode.
CAN_NormalMode_Tx	Demonstrate CAN bus transmit a message with normal mode.
CAN_NormalMode_Tx_Rx	Demonstrate CAN bus transmit and receive a message with normal mode by connecting CAN 0 and CAN1 to the same CAN bus.
CLK_ClockDetector	Show the usage of clock fail detector and clock frequency monitor function.
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.
CRYPTO_AES	Show Crypto IP AES-128 ECB mode encrypt/decrypt function.
CRYPTO_ECC_Demo	ECDSA signature and verification demo
CRYPTO_ECC_ECDH	ECDH demonstrate how to calculate share key by A private key and B private key.
CRYPTO_ECC_GenerateSecretZ	ECDH demo to establish a shared secret key
CRYPTO_ECC_KeyGeneration	Show Crypto IP ECC P-192 key generation function.
CRYPTO_ECC_SignatureGeneration	Show Crypto IP ECC P-192 ECDSA signature generation function.
CRYPTO_ECC_SignatureVerification	Show Crypto IP ECC P-192 ECDSA signature verification function.
CRYPTO_HMAC	Show Crypto IP HMAC function.
CRYPTO_PRNG	Generate random numbers using Crypto IP PRNG.
CRYPTO_SHA	Use Crypto IP SHA engine to run through known answer SHA1 test vectors.
CRYPTO_TDES	Show Crypto IP Triple DES CBC mode encrypt/decrypt function.
DAC_ExtPinTrigger	Demonstrate how to trigger DAC conversion by external pin.
DAC_GroupMode	Demonstrate DAC0 and DAC1 work in group mode
DAC_PDMA_EPWMTrigger	Demonstrate how to use PDMA and trigger DAC0 by EPWM.
DAC_PDMA_TimerTrigger	Demonstrate how to PDMA and trigger DAC by Timer.
DAC_EPWMTrigger	Demonstrate how to trigger DAC by EPWM.
DAC_SoftwareTrigger	Demonstrate how to trigger DAC conversion by software.
DAC_TimerTrigger	Demonstrate how to trigger DAC by timer.

EADC_BandGap	Convert Band-gap (Sample module 16) and print conversion result.
EADC_ADINT_Trigger	Use ADINT interrupt to do the EADC continuous scan conversion.
EADC_PDMA_EPWMTrigger	Demonstrate how to trigger EADC by EPWM and transfer conversion data by PDMA.
EADC_EPWMTrigger	Demonstrate how to trigger EADC by EPWM.
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_SWTRG register.
EADC_TempSensor	Convert temperature sensor (Sample module 17) and print conversion result.
EADC_TimerTrigger	Show how to trigger EADC by timer.
EADC_VBat	Convert VBAT/4 (Sample module 18) and print conversion result.
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.
ECAP_GetInputFreq	Show how to use ECAP interface to get input frequency
ECAP_GetQEIFreq	Show how to use ECAP interface to get QEI frequency.
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 (support KEIL MDK only).
FMC_IAP	Show how to call LDROM function from APROM.
FMC_MultiBoot	Implement a multi-boot system to boot from different applications in APROM. A LDROM code and 4 APROM code are implemented in this sample code.
FMC_OTP	Demonstrate how to program, read and lock OTP.
FMC_ReadAllOne	Demonstrate how to use FMC Read-All-One ISP command to verify APROM/LDROM pages are all 0xFFFFFFFF or not.
FMC_RW	Demonstrate how to read/program embedded Flash by ISP function.
FMC_SecureKey	This sample code shows how to setup the secure key and how to perform secure key comparison.
FMC_XOM	This sample code shows how to configure and setup an XOM region the perform XOM function.
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.
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_MultiBytes_Master	Demonstrate how to use multi-bytes API to access slave. This sample code needs to work with I2C_Slave.
I2C_SingleByte_Master	Demonstrate how to use single byte API to access slave. This sample code needs to work with I2C_Slave.
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.
I2S_Codec	This is an I2S demo using NAU8822/88L25 audio codec, and used to play back the input from line-in.
I2S_Codec_PDMA	This is an I2S demo with PDMA function connected with codec.
I2S_WAVPLAYER	This is a WAV file player which plays back WAV file stored in SD memory card.
OTG_Dual_Role_UMAS	Demonstrate how USB works as a dual role device. If it works as USB Host, it can access a mass storage device. If it works as USB Device, it acts as a mass storage device.
PDMA_BasicMode	Use PDMA0 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).
EPWM_AccumulatorINT_TriggerPDMA	Demonstrate 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_DeadZone	Demonstrate how to use EPWM Dead Zone function.
EPWM_DoubleBuffer	Change duty cycle and period of output waveform by EPWM Double Buffer function (Period loading mode).
EPWM_OutputWaveform	Demonstrate how to use EPWM 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.
QEI_CompareMatch	Show the usage of QEI compare function.
QSPI_DualMode_Flash	Access SPI flash using QSPI dual mode.
QSPI_QuadMode_Flash	Access SPI flash using QSPI quad mode.
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_Dynamic_Tamper	Demonstrate the RTC dynamic tamper function.
RTC_Spare_Access	Demonstrate the RTC spare register read/write function and displays test result to the UART console.
RTC_Static_Tamper	Demonstrate the RTC static tamper function.
RTC_Time_Display	Demonstrate the RTC function and displays current time to the UART console.
SC_EmulateCard	Emulate a smartcard and send ATR to card reader.
SC_ReadATR	Read the smartcard ATR from smartcard 0 interface.
SC_ReadSIM_PhoneBook	Demonstrate how to read phone book information in the SIM card.

SC_Timer	Demonstrate how to use SC embedded timer.
SCUART_TxRx	Demonstrate smartcard UART mode by connecting PB.4 and PB.5 pins.
SDH_FATFS	Access a SD card formatted in FAT file system
SPI_Flash	Access SPI flash through SPI interface.
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.
SPII2S_Master	Configure SPI0 as I2S Master mode and demonstrate how I2S works in Master mode. This sample code needs to work with SPII2S_Slave.
SPII2S_PDMA_Codec	This is an I2S demo with PDMA function connected with audio codec.
SPII2S_PDMA_Play	This is an I2S demo for playing data and demonstrate how I2S works with PDMA.
SPII2S_PDMA_PlayRecord	This is an I2S demo for playing and recording data with PDMA function.
SPII2S_PDMA_Record	This is an I2S demo for recording data and demonstrate how I2S works with PDMA.
SPII2S_Slave	Configure SPI0 as I2S Slave mode and demonstrate how I2S works in Slave mode. This sample code needs to work with SPII2S_Master.

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_TrimIRC	Demonstrate how to use LXT to trim HIRC.
TIMER_ACMPTrigger	Use ACMP to trigger Timer0 counter reset mode.
TIMER_CaptureCounter	Show how to use the timer2 capture function to capture timer2 counter value.
TIMER_Delay	Demonstrate the usage of TIMER_Delay() API to generate a 1 second delay.
TIMER_EventCounter	Demonstrates the timer event counter function.
TIMER_FreeCountingMode	Use the timer0 pin PA.11 to demonstrate timer free counting mode function. And displays the measured input frequency to UART console.
TIMER_InterTimerTriggerMode	Use the timer pin PB.5 to demonstrate inter timer trigger mode function. Also 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_Brake	Demonstrate how to use Timer0 PWM brake function..
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 timer to wake up system from Power-down mode periodically
TIMER_ToggleOut	Demonstrate the timer0 toggle out function on pin PB.5.

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_LIN	Transmit LIN frame including header and response in UART LIN 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.
USBD_Audio_Codec	Demonstrate how to implement a USB audio class device.
USBD_HID_Keyboard	Demonstrate how to implement a USB keyboard device. It 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_MouseKeyboard	Simulate an USB HID mouse and HID keyboard. Mouse draws circle on the screen and Keyboard use GPIO to simulate key input.
USBD_HID_RemoteWakeup	Demonstrate how to implement a USB mouse device. It use PA0 ~ PA5 to control mouse direction and mouse key. It also supports USB suspend and remote wakeup.
USBD_HID_Touch	Demonstrate how to implement a USB touch digitizer device. Two lines demo in Paint.
USBD_HID_Transfer	Demonstrate how to 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_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 a 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_HID_Transfer_CTRL	Use USB Host core driver and HID driver. It shows how to submit HID class request and how to read data from control pipe. A windows tool is also included in this sample code to connect with a USB device.
USBD_Mass_Storage_CDROM	Demonstrate the emulation of USB Mass Storage Device CD-ROM.
USBD_Mass_Storage_SRAM	Use internal SRAM as back end storage media to simulate a 44 KB USB pen drive.
USBD_Micro_Printer	Demonstrate 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	Demonstrate how to implement a composite device (VCOM and HID Keyboard).
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_DualPort	Demonstrate how to implement a USB dual virtual COM port device.
USBD_VCOM_SerialEmulator	Demonstrate how to implement a USB virtual COM port device.
USBH_AudioClass	Demonstrate how to use USBH Audio Class driver. It shows the mute, volume, auto-gain, channel and sampling rate control.
USBH_DEV_CONN	Use connect/disconnect callback functions to handle of device connect and disconnect events.

USBH_HID	Use USB Host core driver and HID driver. This sample demonstrates how to submit HID class request and how to read data from interrupt pipe. This sample supports dynamic device plug/un-plug and multiple HID devices.
USBH_HID_Keybaord	Demonstrate reading key inputs from USB keyboards. This sample includes an USB keyboard driver which is based on the HID driver.
USBH_MassStorage	This sample uses a command-shell-like interface to demonstrate how to use USBH mass storage driver and make it working as a disk driver under FATFS file system.
USBH_UAC_HID	This sample shows how to use USBH Audio Class driver and HID driver at the same time. The target device is a Game Audio (UAC+HID composite device).
USBH_UAC_LoopBack	Receives audio data from UAC device, and immediately send back to that UAC device.
USBH_VCOM	Demonstrates how to use the USB Host core driver and CDC driver to connect a CDC class VCOM device.
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 access 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_MultiBytes_Master	Demonstrate how to use multi-bytes API to access slave. This sample code needs to work with USCI_I2C_Slave.
USCI_I2C_SingleByte_Master	Demonstrate how to use single byte API to access slave. This sample code needs to work with USCI_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.
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 sample code.
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_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 sample code.
USCI_UART_AutoBaudRate	Show how to use auto baud rate detection function.
USCI_UART_Autoflow	Transmit and receive data using auto flow control.
USCI_UART_PDMA	Transmit and receive UART data with PDMA.
USCI_UART_RS485	Transmit and receive data in RS485 mode.
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.
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.
WWDT_CompareINT	Show how to reload the WWDT counter value.

11 .\SampleCode\XOM

XOMLib	Demonstrate how to create XOM library.
XOMLibDemo	Demonstrate how to use XOMLib.

12 REVISION HISTORY

Date	Revision	Description
2019.03.29	3.00.001	Initially issued.
2019.11.07	3.00.002	Add ISP sample codes

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.