

## M480 Series BSP Directory

Directory Introduction for 32-bit NuMicro® Family

### Directory Information

<b>Document</b>	Driver reference manual and revision history.
<b>Library</b>	Driver header and source files.
<b>SampleCode</b>	Driver sample code.
<b>ThirdParty</b>	Library from third party, including FatFs, LibMAD, lwIP, uIP, FreeRTOS™, and Mbed TLS.

*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](http://www.nuvoton.com)

## 1 Document Information

<b>CMSIS.html</b>	Document of CMSIS version 4.5.0.
<b>NuMicro M480 CMSIS BSP Revision History.pdf</b>	This document shows the revision history of M480 BSP.
<b>NuMicro M480 Driver Reference Guide.chm</b>	This document describes the usage of drivers in M480 BSP.

## 2 Library Information

<b>CMSIS</b>	Cortex <sup>®</sup> Microcontroller Software Interface Standard (CMSIS) V4.5.0 definitions by ARM <sup>®</sup> Corp.
<b>Device</b>	CMSIS compliant device header file.
<b>SmartcardLib</b>	Smartcard library binary and header file.
<b>StdDriver</b>	All peripheral driver header and source files.
<b>UsbHostLib</b>	USB host library source code.

### 3 Sample Code Information

<b>CortexM4</b>	Cortex <sup>®</sup> -M4 sample code.
<b>CRYPTO_MbedTLS</b>	Mbed TLS test suites using Crypto accelerator.
<b>FreeRTOS</b>	Simple FreeRTOS <sup>™</sup> demo code.
<b>FreeRTOS_lwIP</b>	FreeRTOS <sup>™</sup> + lwIP sample code.
<b>Hard_Fault_Sample</b>	Show hard fault information when hard fault happened.
<b>Heart_beating</b>	Measure heartbeat rate by amplifying and filtering electrocardiogram signals through OPA, and converting them into digit values through ADC. The calculated heartbeat rate will be sent and printed on screen by UART.
<b>SecureBoot</b>	Firmware update samples in secure boot mode.
<b>Semihost</b>	Show how to debug with semi-host message print.
<b>StdDriver</b>	Sample code to demonstrate the usage of M480 MCU peripheral driver APIs.
<b>Template</b>	A project template for M480.

## 4 Third Party Information

<b>FatFs</b>	A generic FAT file system module for small embedded systems. Its official website is: <a href="http://elm-chan.org/fsw/ff/00index_e.html">http://elm-chan.org/fsw/ff/00index_e.html</a> .
<b>FreeRTOS</b>	A real time operating system available for free download. Its official website is: <a href="http://www.freertos.org/">http://www.freertos.org/</a> .
<b>LibMAD</b>	A MPEG audio decoder library which currently supports MPEG-1 and the MPEG-2 extension to lower sampling frequencies, as well as the de facto MPEG 2.5 format. All three audio layers — Layer I, Layer II, and Layer III (i.e. MP3) are fully implemented. This library is distributed under GPL license. Please contact Underbit Technologies ( <a href="http://www.underbit.com/">http://www.underbit.com/</a> ) for the commercial license.
<b>lwIP</b>	A widely used open source TCP/IP stack designed for embedded systems. Its official website is: <a href="http://savannah.nongnu.org/projects/lwip/">http://savannah.nongnu.org/projects/lwip/</a> .
<b>mbedtls-2.6.0</b>	A portable, easy to use, readable and flexible SSL library. Mbed TLS is available as open source under the Apache 2.0 license or the GPL 2.0 license. Its official website: <a href="https://tls.mbed.org/">https://tls.mbed.org/</a>
<b>uip-0.9</b>	uIP is a very small implementation of the TCP/IP stack that is written by Adam Dunkels < <a href="mailto:adam@sics.se">adam@sics.se</a> >. More information can be obtained from the uIP homepage at <a href="http://www.sics.se/~adam/uip/">http://www.sics.se/~adam/uip/</a> .

## **5 \SampleCode\CortexM4**

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

## 6 \SampleCode\FreeRTOS\_lwIP

<b>httpd</b>	A simple HTTP server demonstrates LwIP under FreeRTOS™. This HTTP server's IP address could be configured statically to 192.168.0.2, or assign by DHCP server.
<b>TCP_EchoServer</b>	A TCP echo server which is implemented with LwIP under FreeRTOS™. This echo server listens to port 80, and its IP address could be configured statically to 192.168.1.2 or assigned by DHCP server. This server replies "Hello World!!" if the received string is "nuvoton", otherwise replies "Wrong Password!!" to its client.
<b>UDP_EchoServer</b>	A UDP echo server which is implemented with LwIP under FreeRTOS™. This echo server listens to port 80, and its IP address could be configured statically to 192.168.1.2 or assigned by DHCP server. After receiving any string from its peer, this server echoes that string back.

## 7 \SampleCode\SecureBoot

<b>HSUSBD_FWUpdate</b>	Use MKROM API to update firmware. This sample code is executed in APROM, and uses MKROM API to update LDROM. After system reset, the program will boot from LDROM.
<b>HSUSBD_IAP</b>	Use MKROM API to do IAP. This sample code is executed in APROM, and loads an image to LDROM to do Secure boot ISP to update APROM. Remember to reset the system after firmware update is complete.
<b>OTA_FWUpdate</b>	Use MKROM API to update firmware. This sample code is executed in LDROM, and uses it to boot from APROM or update APROM.
<b>UART_FWUpdate</b>	Use MKROM API to update firmware. This sample code is executed in LDROM, and uses MKROM API to update APROM. After system reset, the program will boot from APROM.



## 8 \SampleCode\StdDriver

<b>ACMP_ComapreDAC</b>	Demonstrate analog comparator (ACMP) comparison by comparing ACMP0_P0 input and DAC voltage and shows the result on UART console.
<b>ACMP_ComapreVBG</b>	Demonstrate analog comparator (ACMP) comparison by comparing ACMP0_P0 input and VBG voltage and shows the result on UART console.
<b>ACMP_Wakeup</b>	Use ACMP to wake up system from Power-down mode while comparator output changes.
<b>ACMP_WindowCompare</b>	Show how to monitor ACMP input with window compare function.
<b>ACMP_WindowLatch</b>	Demonstrate how to use ACMP window latch mode.
<b>BPWM_Capture</b>	Use BPWM0 Channel 0(PA.0) to capture the BPWM1 Channel 0(PE.13) Waveform
<b>BPWM_DoubleBuffer</b>	Change duty cycle and period of output waveform by BPWM Double Buffer function.
<b>BPWM_OutputWaveform</b>	Demonstrate how to use BPWM counter output waveform.
<b>BPWM_SwitchDuty</b>	Change duty cycle of output waveform by configured period.
<b>BPWM_SyncStart</b>	Demonstrate how to use BPWM counter synchronous start function.
<b>CAN_BasicMode_Rx</b>	Demonstrate CAN bus receive a message with basic mode.
<b>CAN_BasicMode_Tx</b>	Demonstrate CAN bus transmit a message with basic mode.
<b>CAN_BasicMode_Tx_Rx</b>	Demonstrate CAN bus transmit and receive a message with basic mode by connecting CAN0 and CAN1 to the same CAN bus.
<b>CAN_NormalMode_Rx</b>	Demonstrate CAN bus receive a message with normal mode.

<b>CAN_NormalMode_Tx</b>	Demonstrate CAN bus transmit a message with normal mode.
<b>CAN_NormalMode_Tx_Rx</b>	Demonstrate CAN bus transmit and receive a message with normal mode by connecting CAN 0 and CAN1 to the same CAN bus.
<b>CLK_ClockDetector</b>	Demonstrate the usage of clock fail detector and clock frequency range detector function.
<b>CRC_CCITT</b>	Implement CRC in CRC-CCITT mode and get the CRC checksum result.
<b>CRC_CRC8</b>	Implement CRC in CRC-8 mode and get the CRC checksum result.
<b>CRC_CRC32</b>	Implement CRC in CRC-32 mode and get the CRC checksum result.
<b>CRYPTO_AES</b>	Show Crypto IP AES-128 ECB mode encrypt/decrypt function.
<b>CRYPTO_ECC_GenerateSecretZ</b>	Show Crypto IP ECC CDH secret Z generation.
<b>CRYPTO_ECC_KeyGeneration</b>	Show Crypto IP ECC P-192 key generation function.
<b>CRYPTO_ECC_Signature Generation</b>	Show Crypto IP ECC P-192 ECDSA signature generation function.
<b>CRYPTO_ECC_Signature Verification</b>	Show Crypto IP ECC P-192 ECDSA signature verification function.
<b>CRYPTO_HMAC</b>	Show Crypto IP HMAC function.
<b>CRYPTO_PRNG</b>	Generate random numbers using Crypto IP PRNG.
<b>CRYPTO_SHA</b>	Use Crypto IP SHA engine to run through known answer SHA1 test vectors.
<b>CRYPTO_TDES</b>	Show Crypto IP Triple DES CBC mode encrypt/decrypt function.

<b>DAC_EPWMTrigger</b>	Demonstrate EPWM trigger DAC to convert sine wave outputs.
<b>DAC_ExtPinTrigger</b>	Demonstrate external pin trigger DAC convert sine wave outputs.
<b>DAC_GroupMode</b>	Show how to make 2 DAC output channels work in group mode.
<b>DAC_PDMA_EPWMTrigger</b>	Show EPWM trigger DAC to fetch data with PDMA and convert sine wave outputs.
<b>DAC_PDMA_TimerTrigger</b>	Show timer trigger DAC to fetch data with PDMA and convert sine wave outputs.
<b>DAC_SoftwareTrigger</b>	Demonstrate software trigger DAC to convert sine wave outputs.
<b>DAC_TimerTrigger</b>	Demonstrate timer trigger DAC to convert sine wave outputs.
<b>EADC_ADINT_Trigger</b>	Use ADINT interrupt to do the EADC continuous scan conversion.
<b>EADC_BandGap</b>	Convert Band-gap (Sample module 16) and print conversion result.
<b>EADC_EPWM_Trigger</b>	Demonstrate how to trigger EADC by EPWM.
<b>EADC_PDMA_EPWM_Trigger</b>	Demonstrate how to trigger EADC by EPWM and transfer conversion data by PDMA.
<b>EADC_Pending_Priority</b>	Demonstrate how to trigger multiple sample modules and got conversion results in order of priority.
<b>EADC_ResultMonitor</b>	Monitor the conversion result of channel 2 by the digital compare function.
<b>EADC_SWTRG_Trigger</b>	Trigger EADC by writing EADC_SWTRG register.
<b>EADC_TempSensor</b>	Convert temperature sensor (Sample module 17) and print conversion result.
<b>EADC_Timer_Trigger</b>	Show how to trigger EADC by timer.

<b>EADC_VBat</b>	Convert VBAT/4 (Sample module 18) and print conversion result.
<b>EBI_NOR</b>	Configure EBI interface to access NOR Flash connects on EBI interface.
<b>EBI_SRAM</b>	Configure EBI interface to access SRAM connects on EBI interface.
<b>ECAP_GetInputFreq</b>	Show how to use ECAP to measure clock frequency.
<b>ECAP_GetQEIFreq</b>	Show how to use ECAP interface to get QEIA frequency.
<b>EMAC_Iwiperf</b>	A LwIP iperf speed test sample in M480.
<b>EMAC_TimeStamp</b>	Demonstrate the usage of Ethernet time stamp function. It sets current time to 1000 second and prints out current time every second. It also sets an alarm at 1010 second. And rewind current time by 5 seconds after the alarm.
<b>EMAC_TxRx</b>	This Ethernet sample tends to get a DHCP lease from DHCP server, and use 192.168.10.10 as IP address if failed to get a lease. After IP address configured, this sample can reply to PING packets.
<b>EMAC_uIP_httpd</b>	Implement a HTTP server using uIP.
<b>EMAC_uIP_telnetd</b>	Implement a Telnet server using uIP.
<b>EPWM_AccumulatorINT_TriggerPDMA</b>	Demonstrate EPWM accumulator interrupt trigger PDMA.
<b>EPWM_Brake</b>	Demonstrate how to use EPWM brake function.
<b>EPWM_Capture</b>	Capture the EPWM1 Channel 0 waveform by EPWM1 Channel 2.
<b>EPWM_DeadTime</b>	Demonstrate how to use EPWM Dead Time function.
<b>EPWM_DoubleBuffer</b>	Change duty cycle and period of output waveform by EPWM Double Buffer function.
<b>EPWM_OutputWaveform</b>	Demonstrate how to use PWM output waveform.

<b>EPWM_PDMA_Capture</b>	Capture the EPWM1 Channel 0 waveform by EPWM1 Channel 2, and use PDMA to transfer captured data.
<b>EPWM_SwitchDuty</b>	Change duty cycle of output waveform by configured period.
<b>EPWM_SyncStart</b>	Demonstrate how to use PWM counter synchronous start function.
<b>FMC_CRC32</b>	Demonstrate how to use FMC CRC32 ISP command to calculate the CRC32 checksum of APROM, LDROM, and SPROM.
<b>FMC_Dual_Bank</b>	Show FMC dual bank capability. Non-blocking program APROM bank1 while program is running on APROM bank0 is running on bank0 without being blocked
<b>FMC_ExecInSRAM</b>	Implement a code and execute in SRAM to program embedded Flash.
<b>FMC_IAP</b>	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.
<b>FMC_MultiBoot</b>	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.
<b>FMC_MultiWordProgram</b>	Show FMC multi-word program ISP command to program APROM 0x00000~0x20000 area.
<b>FMC_OTP</b>	Demonstrate how to program, read, and lock OTP.
<b>FMC_ReadAllOne</b>	Demonstrate how to use FMC Read-All-One ISP command to verify APROM/LDROM pages are all 0xFFFFFFFF or not.
<b>FMC_RW</b>	Show FMC read Flash IDs, erase, read, and write functions.
<b>FMC_SecureKey</b>	Show how to setup the KPROM and how to perform KPROM comparison.

<b>FMC_SPROM</b>	Show how to make an application running on APROM but with a sub-routine on SPROM, which can be secured
<b>GPIO_EINTAndDebounce</b>	Show the usage of GPIO external interrupt function and de-bounce function.
<b>GPIO_INT</b>	Show the usage of GPIO interrupt function.
<b>GPIO_OutputInput</b>	Show how to set GPIO pin mode and use pin data input/output control.
<b>GPIO_PowerDown</b>	Show how to wake up system from Power-down mode by GPIO interrupt.
<b>HSOTG_Dual_Role_UMAS</b>	An OTG sample code which will become a USB host when connected with a Micro-A cable, and can access the pen drive when plugged in. It will become a removable disk when connected with a Micro-B cable, and then plug into PC.
<b>HSOTG_HNP</b>	Show HID mouse with OTG HNP protocol.
<b>HSUSBD_Audio10_Codec</b>	An UAC1.0 sample used to record and play the sound sent from PC through the USB interface
<b>HSUSBD_Audio20_Codec</b>	An UAC2.0 sample used to record and play the sound sent from PC through the USB interface.
<b>HSUSBD_HID_Mouse</b>	Simulate a USB mouse and draws circle on the screen.
<b>HSUSBD_HID_MouseKeyboard</b>	Simulate a USB mouse and a USB keyboard.
<b>HSUSBD_HID_Transfer</b>	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.
<b>HSUSBD_Mass_Storage_DataFlash</b>	Use embedded Data Flash as storage to implement a USB Mass-Storage device.
<b>HSUSBD_Mass_Storage_SactterGather</b>	Demonstrate the usage of USB DMA scatter gather function.
<b>HSUSBD_Mass_Storage_</b>	Implement a mass storage class sample to demonstrate

<b>ShortPacket</b>	how to receive a USB short packet.
<b>HSUSBD_Mass_Storage_SRAM</b>	Use internal SRAM as back end storage media to simulate a 30 KB USB pen drive.
<b>HSUSBD_VCOM_SerialEmulator</b>	Demonstrate how to implement a USB virtual com port device.
<b>HSUSBH_USBH_AudioClass</b>	Demonstrate how to use USBH Audio Class driver. It shows the mute, volume, auto-gain, channel, and sampling rate control.
<b>HSUSBH_USBH_DEV_CONN</b>	Use connect/disconnect callback functions to handle of device connect and disconnect events.
<b>HSUSBH_USBH_Firmware_Update</b>	Automatically search and read new firmware from USB drive, if found, update APROM Flash with it.
<b>HSUSBH_USBH_HID</b>	Use USB Host core driver and HID driver. This sample demonstrates how to submit HID class request and read data from interrupt pipe. This sample supports dynamic device plug/un-plug and multiple HID devices.
<b>HSUSBH_USBH_HID_Keyboard</b>	Demonstrate reading key inputs from USB keyboards. This sample includes an USB keyboard driver which is based on the HID driver.
<b>HSUSBH_USBH_MassStorage</b>	Use a command-shell-like interface to demonstrate how to use USBH mass storage driver and make it work as a disk driver under the FATFS file system.
<b>HSUSBH_USBH_SPIM_Writer</b>	Provide a command line interface for reading files from USB disk and writing to SPIM Flash. This sample code also provides functions of dump SPIM Flash, compares USB disk file with SPIM Flash, and branches to run code on SPIM.
<b>HSUSBH_USBH_UAC_HID</b>	Show 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).
<b>HSUSBH_USBH_UAC_LoopBack</b>	Receives audio data from UAC device, and immediately send back to that UAC device.
<b>HSUSBH_USBH_VCOM</b>	Demonstrates how to use the USB Host core driver and



	CDC driver to connect a CDC class VCOM device.
<b>I2C_EEPROM</b>	Read/write EEPROM via I <sup>2</sup> C interface.
<b>I2C_Loopback</b>	Demonstrate how a Master accesses Slave.
<b>I2C_Master</b>	An I <sup>2</sup> C master mode demo code.
<b>I2C_Master_PDMA</b>	Demonstrate how a Master accesses Slave using PDMA TX mode and PDMA RX mode.
<b>I2C_MultiBytes_Master</b>	Demonstrate how to use multi-bytes API to access slave. This sample code needs to work with I2C_Slave.
<b>I2C_PDMA</b>	Demonstrate I2C PDMA mode which needs to connect I2C0 (Master) and I2C1 (Slave).
<b>I2C_SingleByte_Master</b>	Demonstrate how to use single byte API to access slave. This sample code needs to work with I2C_Slave.
<b>I2C_Slave</b>	An I <sup>2</sup> C slave mode demo code.
<b>I2C_Slave_PDMA</b>	Demonstrate how a Slave uses PDMA Rx mode receive data from a Master.
<b>I2C_Wakeup_Slave</b>	Demonstrate how to set I2C to wake up MCU from Power-down mode. This sample code needs to work with I2C_Master.
<b>I2S_Codec</b>	An I <sup>2</sup> S demo used to play back the input from line-in or MIC interface.
<b>I2S_Codec_PDMA</b>	An I <sup>2</sup> S with PDMA demo used to play back the input from line-in or MIC interface.
<b>I2S_MP3PLAYER</b>	A MP3 player sample which plays MP3 files stored on SD memory card.
<b>I2S_WAVPLAYER</b>	A WAV file player which plays back WAV file stored in a USB pen drive.
<b>OPA_Control</b>	Show how to control OPA.
<b>OTG_Dual_Role_UMAS</b>	An OTG sample code which will become a USB host when connected with a Micro-A cable, and can access the



	pen drive when plugged in. It will become a removable disk when connected with a Micro-B cable, and then plug into PC.
<b>OTG_HNP</b>	Show HID mouse with OTG HNP protocol.
<b>PDMA_BasicMode</b>	Use PDMA channel 2 to demonstrate memory to memory transfer.
<b>PDMA_ScatterGather</b>	Use PDMA channel 5 to demonstrate memory to memory transfer by scatter-gather mode.
<b>PDMA_ScatterGather_PingPongBuffer</b>	Use PDMA to implement Ping-Pong buffer by scatter-gather mode (memory to memory).
<b>QEI_CompareMatch</b>	Show the usage of QEI compare function.
<b>SPI_DualMode_Flash</b>	Access SPI Flash using QSPI dual mode.
<b>SPI_QuadMode_Flash</b>	Access SPI Flash using QSPI quad mode.
<b>RTC_Alarm_Test</b>	Demonstrate the RTC alarm function. It sets an alarm 10 seconds after execution.
<b>RTC_Alarm_Wakeup</b>	Use RTC alarm interrupt event to wake up system.
<b>RTC_Dynamic_Tamper</b>	Show how to use RTC dynamic tamper function.
<b>RTC_Spare_Access</b>	Show how to access RTC spare registers.
<b>RTC_Static_Tamper</b>	Show how to use RTC static tamper function.
<b>RTC_Time_Display</b>	Demonstrate the RTC function and displays current time to the UART console.
<b>SC_ReadATR</b>	Read the smartcard ATR from smartcard 1 interface.
<b>SC_ReadSIM_PhoneBook</b>	Demonstrate how to read phone book information in the SIM card.
<b>SC_Timer</b>	Demonstrate how to use SC embedded timer
<b>SCUART_TxRx</b>	Demonstrate smartcard UART mode by connecting PA.0 and PA.1 pins.

<b>SDH_FATFS</b>	Access a SD card formatted in FAT file system.
<b>SDH_FATFS_Dual</b>	Access two SD cards formatted in FAT file system.
<b>SDH_Firmware_Update</b>	Automatically search and read new firmware from SD card, if found, update APROM Flash with it.
<b>SPI_Flash</b>	Access SPI Flash through SPI interface.
<b>SPI_LoopBack</b>	A SPI read/write demo connecting QSPI0 MISO and MOSI pins.
<b>SPI_MasterFIFOmode</b>	Configure QSPI0 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.
<b>SPI_PDMA_LoopTest</b>	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.
<b>SPI_SlaveFIFOmode</b>	Configure QSPI0 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.
<b>SPII2S_Master</b>	Configure SPI1 as I <sup>2</sup> S Master mode and demonstrate how I <sup>2</sup> S works in Master mode.
<b>SPII2S_PDMA_Codec</b>	An I <sup>2</sup> S demo with PDMA function connected with audio codec.
<b>SPII2S_PDMA_Play</b>	An I <sup>2</sup> S demo for playing data and demonstrating how I <sup>2</sup> S works with PDMA.
<b>SPII2S_PDMA_PlayRecord</b>	An I <sup>2</sup> S demo for playing and recording data with PDMA function.
<b>SPII2S_PDMA_Record</b>	An I <sup>2</sup> S demo for recording data and demonstrating how I <sup>2</sup> S works with PDMA.
<b>SPII2S_Slave</b>	Configure SPI1 as I <sup>2</sup> S Slave mode and demonstrate how I <sup>2</sup> S works in Slave mode. This sample code needs to work

	with SPII2S_Master.
<b>SPIM_CIPHER</b>	Demonstrate SPIM DMA read/write with cipher enabled. This sample also dumps SPI Flash content via I/O mode read to prove it is encrypted cipher context.
<b>SPIM_DMA_RW</b>	Demonstrate SPIM DMA mode read/write function.
<b>SPIM_DMM</b>	Demonstrate SPIM DMM mode read function. This sample programs SPI Flash with DMA write and verify Flash with DMA read and CPU read respectively.
<b>SPIM_DMM_RUN_CODE</b>	Show how to make an application booting from APROM with a sub-routine resided on SPIM flash.
<b>SPIM_IO_RW</b>	Demonstrate how to issue SPI Flash erase, program, and read commands under SPIM I/O mode.
<b>SYS_BODWakeup</b>	Demonstrate how to wake up system from Power-down mode by brown-out detector interrupt.
<b>SYS_DPDMode_Wakeup</b>	Demonstrate how to wake up system from Deep Power-down mode by Wake-up pin(PC.0) or Wake-up Timer..
<b>SYS_PLLClockOutput</b>	Change system clock to different PLL frequency and output system clock from CLKO pin.
<b>SYS_SPDMode_Wakeup</b>	Demonstrate how to wake up system from Standby Power-down mode by GPIO pin(PA.0), Wake-up Timer, Wake-up ACMP, RTC Tick, RTC Alarm, RTC Tamper 0 , BOD, or LVR.
<b>SYS_TrimIRC</b>	Demonstrate how to use LXT to trim HIRC.
<b>TIMER_ACMPTrigger</b>	Use ACMP to trigger timer reset mode.
<b>TIMER_CaptureCounter</b>	Show how to use the timer2 capture function to capture timer2 counter value.
<b>TIMER_Delay</b>	Demonstrate the usage of TIMER_Delay() API to generate a 1 second delay.
<b>TIMER_EventCounter</b>	Use pin PD.4 to demonstrates timer event counter function.

<b>TIMER_FreeCountingMode</b>	Use the timer pin PA.7 to demonstrate timer free counting mode function. And displays the measured input frequency to UART console.
<b>TIMER_InterTimerTriggerMode</b>	Use the timer pin PD.4 to demonstrate inter-timer trigger mode function. Also display the measured input frequency to UART console.
<b>TIMER_Periodic</b>	Use the timer periodic mode to generate timer interrupt every 1 second.
<b>TIMER_PeriodicINT</b>	Implement timer counting in periodic mode.
<b>TIMER_PWM_Brake</b>	Demonstrate how to use Timer PWM brake function.
<b>TIMER_PWM_ChangeDuty</b>	Change duty cycle and period of output waveform by Timer PWM Double Buffer function.
<b>TIMER_PWM_DeadTime</b>	Demonstrate how to use Timer PWM Dead Time function.
<b>TIMER_PWM_OutputWaveform</b>	Enable 4 Timer PWM output channels with different frequency and duty ratio.
<b>TIMER_ToggleOut</b>	Demonstrate the timer 0 toggle out function on pin PD.4.
<b>TIMER_Wakeup</b>	Use Timer to wake up system from Power-down mode periodically.
<b>UART_AutoBaudRate</b>	Show how to use auto baud rate detection function.
<b>UART_AutoFlow</b>	Transmit and receive data using auto flow control.
<b>UART_IrDA</b>	Transmit and receive UART data in UART IrDA mode.
<b>UART_LIN</b>	Demonstrate how to send data to LIN bus.
<b>UART_PDMA</b>	Demonstrate UART transmit and receive function with PDMA.
<b>UART_RS485</b>	Transmit and receive data in UART RS485 mode.
<b>UART_TxRxFunction</b>	Transmit and receive data from PC terminal through RS232 interface.

<b>UART_Wakeup</b>	Show how to wake up system from Power-down mode by UART interrupt.
<b>USBD_Audio_Codec</b>	Demonstrate how to implement a USB audio class device.
<b>USBD_HID_Keyboard</b>	Demonstrate how to implement a USB keyboard device. This sample code supports to use GPIO to simulate key input.
<b>USBD_HID_Mouse</b>	Simulate a USB mouse and draws circle on the screen.
<b>USBD_HID_MouseKeyboard</b>	Simulate an USB HID mouse and HID keyboard. Mouse draws circle on the screen and Keyboard uses GPIO to simulate key input.
<b>USBD_HID_RemoteWakeup</b>	Simulate a HID mouse supports USB suspend and remote wakeup.
<b>USBD_HID_Touch</b>	Demonstrate how to implement a USB touch digitizer device. Two lines demo in Paint.
<b>USBD_HID_Transfer</b>	Demonstrate how to 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.
<b>USBD_HID_Transfer_And_Keyboard</b>	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.
<b>USBD_HID_Transfer_And_MSC</b>	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.
<b>USBD_HID_Transfer_CTRL</b>	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.
<b>USBD_Mass_Storage_</b>	Demonstrate the emulation of USB Mass Storage Device

<b>CDFROM</b>	CD-ROM.
<b>USBD_Mass_Storage_Flash</b>	Use internal Flash as back end storage media to simulate a USB pen drive.
<b>USBD_Mass_Storage_SRAM</b>	Use internal SRAM as back end storage media to simulate a 30 KB USB pen drive.
<b>USBD_Micro_Printer</b>	Demonstrate how to implement a USB micro printer device.
<b>USBD_Printer_And_HID_Transfer</b>	Demonstrate how to implement a composite device. (USB micro printer device and 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 a USB device.
<b>USBD_VCOM_And_HID_Keyboard</b>	Demonstrate how to implement a composite device.(VCOM and HID keyboard)
<b>USBD_VCOM_And_HID_Transfer</b>	Demonstrate how to implement a composite device.(VCOM and 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 a USB device.
<b>USBD_VCOM_And_Mass_Storage</b>	Demonstrate how to implement a composite device.(Virtual COM port and Mass storage device)
<b>USBD_VCOM_DualPort</b>	Demonstrate how to implement a USB dual virtual COM port device.
<b>USBD_VCOM_SerialEmulator</b>	Demonstrate how to implement a USB virtual COM port device.
<b>USCI_I2C_EEPROM</b>	Show how to use USCI_I2C interface to access EEPROM.
<b>USCI_I2C_Lookback</b>	Show a Master how to access 7-bit address Slave (loopback).
<b>USCI_I2C_Loopback_10bit</b>	Show a Master how to access 10-bit address Slave (loopback).
<b>USCI_I2C_Master</b>	Show a Master how to access Slave.

<b>USCI_I2C_Master_10bit</b>	Show a Master how to access 10-bit address Slave.
<b>USCI_I2C_Monitor</b>	Use USCI_I2C to monitor and log I <sup>2</sup> C bus traffic.
<b>USCI_I2C_MultiBytes_Master</b>	Demonstrate how to use multi-bytes API to access slave. This sample code needs to work with USCI_I2C_Slave.
<b>USCI_I2C_SingleByte_Master</b>	Demonstrate how to use single byte API to access slave. This sample code needs to work with USCI_I2C_Slave.
<b>USCI_I2C_Slave</b>	Show a Slave how to receive data from Master.
<b>USCI_I2C_Slave_10bit</b>	Show a 10-bit address Slave how to receive data from Master.
<b>USCI_I2C_Wakeup_Slave</b>	Show how to wake-up USCI_I2C from deep sleep mode.
<b>USCI_SPI_Loopback</b>	A USCI_SPI read/write demo connecting SPI0 and SPI1 interface.
<b>USCI_SPI_MasterMode</b>	Configure USCI_SPI1 as Master mode and demonstrate how to communicate with an off-chip SPI Slave device. Needs to work with USCI_SPI_SlaveMode sample code.
<b>USCI_SPI_PDMA_LoopTest</b>	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.
<b>USCI_SPI_SlaveMode</b>	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.
<b>USCI_UART_AutoBaudRate</b>	Show how to use auto baud rate detection function.
<b>USCI_UART_Autoflow_Master</b>	Transmit and receive data with auto flow control. This sample code needs to work with USCI_UART_Autoflow_Slave.
<b>USCI_UART_Autoflow_Slave</b>	Transmit and receive data with auto flow control. This sample code needs to work with USCI_UART_Autoflow_Master.



<b>USCI_UART_PDMA</b>	Demonstrate UART data transfer with PDMA.
<b>USCI_UART_RS485_Master</b>	Transmit and receive data in RS485 mode. This sample code needs to work with USCI_UART_RS485_Slave.
<b>USCI_UART_RS485_Slave</b>	Transmit and receive data in RS485 mode. This sample code needs to work with USCI_UART_RS485_Master.
<b>USCI_UART_TxRxFunction</b>	Transmit and receive data from PC terminal through RS232 interface.
<b>USCI_UART_Wakeup</b>	Show how to wake up system from Power-down mode by USCI interrupt in UART mode.
<b>WDT_TimeoutWakeupAndReset</b>	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.
<b>WWDT_CompareINT</b>	Show how to reload the WWDT counter value.



### **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.*