

## NUC472/NUC442 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, LibMAD, and FreeRTOS™.

*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

CMSIS.html	Document of CMSIS version 4.5.0
NuMicro NUC472_NUC442 Series CMSIS BSP Revision History.pdf	This document shows the revision history of NUC472/NUC442 BSP.
NuMicro NUC472 NUC442 Driver Reference Guide.html	This document describes the usage of drivers in NUC472/NUC442 BSP.

## 2 Library Information

<b>CMSIS</b>	Cortex® Microcontroller Software Interface Standard (CMSIS) V4.5.0 definitions by ARM® 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®-M4 sample code.
<b>FreeRTOS</b>	Simple FreeRTOS™ demo code.
<b>FreeRTOS_lwIP_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>FreeRTOS_lwIP_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>FreeRTOS_lwIP_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.
<b>Hard_Fault_Sample</b>	Show hard fault information when hard fault happened.
<b>ISP</b>	
<b>ISP_Updater</b>	Sample ISP updater that reads firmware from pen driver and updates to APROM.
<b>NUC472-NuTiny</b>	Sample code for NUC472 Tiny Board.
<b>RTX_Blinky</b>	Simple RTX demo code that blinks the on-board LED.
<b>Semihost</b>	Show how to debug with semi-host message print.
<b>StdDriver</b>	Sample code to demonstrate the usage of NUC472/NUC442 MCU peripheral driver APIs.
<b>Template</b>	A project template for NUC472/NUC442 MCU.

## 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>FreeRTOSV8.2.1</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-1.4.1</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>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®-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®-M4 MPU.

## 6 \SampleCode\ISP

ISP_CAN	Sample ISP firmware communicated with ISP tool through an CAN interface.
ISP_DFU	Sample ISP firmware communicated with ISP tool through USB DFU( Device Firmware Upgrade) class.
ISP_HID	Sample ISP firmware communicated with ISP tool through a USB HID interface.
ISP_I2C	Sample ISP firmware communicated with ISP tool through an I <sup>2</sup> C interface.
ISP_RS485	Sample ISP firmware communicated with ISP tool through a RS485 interface.
ISP_SPI	Sample ISP firmware communicated with ISP tool through a SPI interface.
ISP_UART	Sample ISP firmware communicated with ISP tool through a UART interface.

## **7 \SampleCode\NUC472-NuTiny**

**LED**

Toggle PB.10 to turn on / off the board LED.



## 8 \SampleCode\StdDriver

<b>ACMP</b>	Demonstrate analog comparator (ACMP) comparison by comparing ACMP0_P0 input and VBG voltage and shows the result on UART console.
<b>ADC_Compare</b>	Demonstrate ADC conversion and comparison function by monitoring the conversion result of channel 0.
<b>ADC_ContinuousScan</b>	Convert ADC channel 0, 1, 2 in continuous scan mode and prints conversion result.
<b>ADC_PDMA</b>	Use PDMA channel 1 to move ADC channel 0, 1, 2 converted data to SRAM.
<b>ADC_SingleCycleScan</b>	Convert ADC channel 0, 1, 2 in single cycle scan mode and prints conversion result.
<b>ADC_SingleMode</b>	Convert ADC channel 0 in single mode and prints conversion result.
<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>CAP_MotionDetection</b>	Implement motion detection with image capture interface.
<b>CAP_Packet_DownScale</b>	Use packet format (all the luma and chroma data interleaved) to store captured image from NT99141

	sensor to SRAM.
<b>CAP_Planar_DownScale</b>	Use planar format (all the luma information for a frame, followed by all the information for one chroma channel, and then the information for the other chroma channel) to store captured image from NT99141 sensor to SRAM.
<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>CRYPTO_AES</b>	Show Crypto IP AES-128 ECB mode encrypt/decrypt 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>EADC_ADINT_Trigger</b>	Demonstrate how to use ADINT interrupt to trigger EADC.
<b>EADC_Compare</b>	Demonstrate EADC conversion and comparison function by monitoring the conversion result of sample module 0 channel 2.
<b>EADC_PWM_Trigger</b>	Show how to trigger EADC0 SAMPLE module 0 by PWM channel 0
<b>EADC_SimultaneousMode</b>	Demonstrate EADC0 sample module 2 and EADC1 sample module 2 in simultaneous sampling mode.
<b>EADC_STADC_Trigger</b>	Demonstrate how to trigger EADC by STADC external signal.
<b>EADC_SWTRG_Trigger</b>	Demonstrate how to trigger EADC by writing EADC_SWTRG register.
<b>EADC_Timer_Trigger</b>	Demonstrate how to trigger EADC by timer.
<b>EBI_SRAM</b>	Configure EBI interface to access SRAM connects on EBI

	interface.
<b>ECAP</b>	Demonstrate the Enhanced capture function using PC.5
<b>FMC_ExecInSRAM</b>	Implement a code and execute in SRAM to program embedded Flash. (Support KEIL® MDK Only.)
<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_Brake</b>	Demonstrate the brake function of EPWM0.
<b>EPWM_DeadZone</b>	Demonstrate the dead-zone feature with EPWM0 channel 0 and channel 1.
<b>FMC_MULTI_WORD_PROG</b>	Show FMC ISP multi-word program function. The loader.bin will load fmc_multi_word_prog.bin to SRAM and execute it.
<b>FMC_RW</b>	Show FMC read flash IDs, erase, read, and write functions.
<b>FMC_VECTOR_REMAP</b>	Show how to branch programs between LDROM, APROM start page, and APROM other page.
<b>GPIO</b>	Use GPIO driver to control the GPIO pin direction, control their high/low state, and how to use GPIO interrupts.
<b>I2C_EEPROM</b>	Read/write EEPROM via I <sup>2</sup> C interface.
<b>I2C_GSENSOR</b>	Read G-sensor (DMARD08) data via I <sup>2</sup> C interface.
<b>I2C_Master</b>	An I <sup>2</sup> C master mode demo code.

<b>I2C_Slave</b>	An I <sup>2</sup> C slave mode demo code.
<b>I2S_MP3PLAYER</b>	A MP3 player sample which plays MP3 files stored on SD memory card.
<b>I2S_NAU8822</b>	An I <sup>2</sup> S demo using NAU8822 audio codec, and used to play back the input from line-in or MIC interface.
<b>I2S_NAU8822_PDMA</b>	An I <sup>2</sup> S with PDMA demo using NAU8822 audio codec, and used to play back the input from line-in or MIC interface.
<b>I2S_WAVPLAYER</b>	A WAV file player which plays back WAV file stored in USB pen drive.
<b>PDMA</b>	Use PDMA channel 2 to demonstrate memory to memory transfer.
<b>PDMA_Scatter_Gather</b>	Use PDMA channel 5 to demonstrate memory to memory transfer by scatter-gather mode.
<b>PS2</b>	Simulate the behavior of a PS/2 mouse by moving the cursor on the screen.
<b>PWM_Capture</b>	Demonstrate PWM Capture function by using PWM0 channel 2 to capture the output of PWM0 channel 0. Please connect PA.5 and PC.10 to execute this code.
<b>PWM_DeadZone</b>	Demonstrate the dead-zone feature with PWM0.
<b>RTC_Alarm_Test</b>	Demonstrate the RTC alarm function. It sets an alarm 10 seconds after execution.
<b>RTC_Spare_Access</b>	Show how to access RTC spare registers.
<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 5 interface.
<b>SCUART_TxRx</b>	Demonstrate smartcard UART mode by connecting PA.7 and PA.10 pins.
<b>SD_FATFS</b>	Access a SD card formatted in FAT file system.

<b>SPI_DualMode_Flash</b>	Access SPI Flash using SPI dual mode.
<b>SPI_Flash</b>	Access SPI Flash through SPI interface.
<b>SPI_LoopBack</b>	A SPI read/write demo by connecting SPI0 and SPI1 interface.
<b>SPI_MasterMode</b>	SPI master mode demo code.
<b>SPI_MasterSlave_PDMA</b>	Demonstrate the usage of PDMA transfer. One SPI interface is use as a host, and the other is slave. Totally 4 PDAM channels are used in this sample.
<b>SPI_QuadMode_Flash</b>	Access SPI Flash using SPI quad mode.
<b>SPI_SlaveMode</b>	SPI slave mode demo code.
<b>SPI_TFT_LCD</b>	Display an image on TFT LCD panel via SPI interface.
<b>SPI_TxRxLoopback_PDMA</b>	Demonstrate the usage of PDMA transfer. One SPI interface is enabled in loopback mode. Two PDMA channels are used in this sample, one for transmit, the other for receive.
<b>SYS_Control</b>	Demonstrate the usage of SYS driver by changing different PLL setting for the system clock source. This sample also enables the CLK0 (PC.5) output with frequency set to system clock / 4.
<b>SYS_PowerDown_MinCurrent</b>	Demonstrate how to minimize power consumption when entering power down mode.
<b>Timer_Delay</b>	Demonstrate the usage of TIMER_Delay() API to generate a 1 second delay.
<b>TIMER_EventCounter</b>	Use pin PB.4 to demonstrates timer event counter function.
<b>Timer_FreeCountingMode</b>	Use the timer pin PC.8 to demonstrate timer free counting mode function. And displays 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_ToggleOut</b>	Demonstrate the timer 0 toggle out function on pin PB.4.
<b>TIMER_Wakeup</b>	Use Timer to wake up system from Power-down mode periodically.
<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_PDMA</b>	Demonstrate UART transmit and receive function with PDMA.
<b>UART_RS485</b>	Transmit and receive data in UART RS485 mode.
<b>UART_TxRx_Function</b>	Transmit and receive data from PC terminal through RS232 interface.
<b>USBD_Audio_Microphone</b>	An UAC1.0 sample used to record the sound to PC through the USB interface.
<b>USBD_Audio_Speaker</b>	An UAC1.0 sample used to play the sound sent from PC through the USB interface.
<b>USBD_Bulk</b>	Sample USB device bulk transfer code.
<b>USBD_HID_MOUSE</b>	Simulate a USB mouse and draws circle on the screen.
<b>USBD_HID_Mouse_Vender</b>	Simulate a USB mouse that supports vendor command.
<b>USBD_HID_MouseKeyboard</b>	Simulate a USB mouse and a USB keyboard.
<b>USBD_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>USBD_Mass_Storage_DataFlash</b>	Use embedded data flash as storage to implement a USB Mass-Storage device.
<b>USBD_Mass_Storage_ShortPacket</b>	Implement a mass storage class sample to demonstrate how to receive a USB short packet.
<b>USBD_Mass_Storage_SRAM</b>	Use internal SRAM as back end storage media to simulate a 30 KB USB pen drive.

<b>USBD_Mass_Storage SactterGather</b>	Demonstrate the usage of USBD DMA scatter gather function.
<b>USBD_VCOM_SerialEmulator</b>	Demonstrate how to implement a USB virtual com port device.
<b>USBD_VENDOR_LBK</b>	A USB device vendor class sample program. This sample code needs to test with USBH_VENDOR_LBK.
<b>USBH_AUDIO_CLASS</b>	A USB Host sample code to support USB Audio Class.
<b>USBH_HID</b>	Use USB Host core driver and HID driver. It shows how to submit HID class request and how to read data from interrupt pipe.
<b>USBH_HID_KEYBOARD</b>	Show how to use USB Host driver to handle HID keyboard devices
<b>USBH_HID_MULTI</b>	Show how to implement a USB Host and recognize multi-HID devices when devices plug-in.
<b>USBH_UAC_HID</b>	A USB Host sample code to support USB Audio Class with HID composite device.
<b>USBH_UMAS</b>	Use USB Host core driver, USB mass storage driver, and FATFS file system to show a disk access shell interface.
<b>USBH_VENDOR_LBK</b>	A USB host vendor class sample program. This sample code needs to test with USBD_VENDOR_LBK.
<b>USBOTG_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>WDT_Polling</b>	Use polling mode to check WDT time-out state and reset WDT after time out occurs.
<b>WDT_Wakeup</b>	Use WDT to wake system up from power-down mode periodically.
<b>WWDT_Reload</b>	Demonstrate the WWDT counter reload function.





### **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.*