

## M4521 Series BSP Directory

Directory Introduction for 32-bit NuMicro® Family

### Directory Information

<b>Document</b>	Driver reference manual and reversion history.
<b>Library</b>	Driver header and source files.
<b>SampleCode</b>	Driver sample code.
<b>ThirdParty</b>	Library from third party

*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>Revision History.pdf</b>	Show all the revision history about specific BSP.
<b>CMSIS.html</b>	Describe all of the information of CMSIS library, including CMSIS-CORE, CMSIS-DSP, CMSIS-RTOS API and CMSIS-SVD.
<b>NuMicro M4521 Driver Reference Guide.chm</b>	Describe the definition, input and output of each API.

## 2 Library Information

<b>CMSIS</b>	CMSIS definitions by ARM® Corp.
<b>Device</b>	CMSIS compliant device header file.
<b>StdDriver</b>	All peripheral driver header and source files.
<b>SmartcardLib</b>	Library for CCID smart card reader.

### 3 Sample Code Information

<b>CardReader</b>	CCID <sup>[1]</sup> smart card reader sample code.
<b>FreeRTOS</b>	Simple FreeRTOS™ demo code.
<b>ISP</b>	ISP firmware samples.
<b>Hard_Fault_Sample</b>	Show hard fault information when hard fault happened.
<b>Template</b>	Software Development Template.
<b>Semihost</b>	Show how to debug with semi-host message print.
<b>RegBased</b>	The sample code able to access control registers directly.
<b>StdDriver</b>	M4521 Series Driver Samples

1. Circuit card interface device (CCID) is USB device that interface with integrated circuit cards.

#### 4 \SampleCode\ISP

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.

## 5 \SampleCode\RegBased

<b>CLK_ClockDetector</b>	Show the usage of clock fail detector and clock frequency monitor 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>EADC_ADINT_Trigger</b>	Use ADINT interrupt to do the ADC continuous scan conversion.
<b>EADC_PDMA_PWM_Trigger</b>	Demonstrate how to trigger EADC by PWM and transfer conversion data by PDMA.
<b>EADC_PWM_Trigger</b>	Demonstrate how to trigger ADC by PWM.
<b>EADC_ResultMonitor</b>	Monitor the conversion result of channel 2 by the digital compare function.
<b>EADC_SWTRG_Trigger</b>	Trigger ADC by writing EADC_SWTRG register.
<b>EADC_Timer_Trigger</b>	Show how to trigger ADC by timer.
<b>EBI_NOR</b>	Configure EBI interface to access MX29LV320T (NOR Flash) on EBI interface.
<b>EBI_SRAM</b>	Configure EBI interface to access BS616LV4017 (SRAM) with PDMA transfer on EBI interface.
<b>FMC_ExecInSRAM</b>	Implement a code and execute in SRAM to program embedded Flash. (Support KEIL® MDK Only)
<b>FMC_IAP</b>	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.
<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_RW</b>	Show how to read/program embedded flash by ISP function.
<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>I2C_EEPROM</b>	Show how to use I <sup>2</sup> C interface to access EEPROM.
<b>I2C_GCMode_Master</b>	Show how a Master uses I <sup>2</sup> C address 0x0 to write data to Slave. This sample code needs to work with I2C_GCMode_Slave.
<b>I2C_GCMode_Slave</b>	Show a Slave how to receive data from Master in GC (General Call) mode. This sample code needs to work with I2C_GCMode_Master.
<b>I2C_Master</b>	Show a Master how to access Slave. This sample code needs to work with I2C_Slave.
<b>I2C_Slave</b>	Show how to set I <sup>2</sup> C in Slave mode and receive the data from Master. This sample code needs to work with I2C_Master.
<b>I2C_SMBus</b>	Show how to control SMBus interface and use SMBus protocol between Host and Slave.
<b>I2C_Wakeup_Master</b>	Show how to wake up MCU from Power-down. This sample code needs to work with I2C_Wakeup_Slave.
<b>I2C_Wakeup_Slave</b>	Show how to wake up MCU from Power-down mode through I <sup>2</sup> C interface. This sample code needs to work with I2C_Wakeup_Master.
<b>I2S_Master</b>	Configure SPI1 as I <sup>2</sup> S Master mode and demonstrate how I <sup>2</sup> S works in Master mode. This sample code needs to work with I2S_Slave sample code.

<b>I2S_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 I2S_Master sample code.
<b>PDMA</b>	Use PDMA channel 2 to transfer data from memory to memory.
<b>PDMA_Scatter_Gather</b>	Use PDMA channel 5 to transfer data from memory to memory by scatter-gather mode.
<b>PDMA_ScatterGather_PingPongBuffer</b>	Use PDMA to implement Ping-Pong buffer by scatter-gather mode (memory to memory).
<b>PWM_Capture</b>	Capture the PWM1 Channel 0 waveform by PWM1 Channel 2.
<b>PWM_DeadZone</b>	Demonstrate how to use PWM Dead Zone function.
<b>PWM_DoubleBuffer</b>	Change duty cycle and period of output waveform by PWM Double Buffer function.
<b>PWM_PDMA_Capture</b>	Capture the PWM1 Channel 0 waveform by PWM1 Channel 2, and use PDMA to transfer captured data.
<b>RTC_AlarmWakeup</b>	Use RTC alarm interrupt event to wake up system.
<b>RTC_SpareRegisterRW</b>	Show how to access RTC spare registers.
<b>RTC_TimeAndTick</b>	Get the current RTC data/time per tick.
<b>SC_ReadATR</b>	Read the smartcard ATR from smartcard 0 interface.
<b>SCUART_TxRx</b>	Show smartcard UART mode by connecting PA.0 and PA.1 pins.
<b>SPI_Loopback</b>	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.
<b>SPI_MasterMode</b>	Configure SPI0 as Master mode and demonstrate how to communicate with an off-chip SPI Slave device. This sample code needs to work with SPI_SlaveMode.



<b>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>SPI_SlaveMode</b>	Configure SPI0 as Slave mode and demonstrate how to communicate with an off-chip SPI Master device. This sample code needs to work with SPI_MasterMode.
<b>SYS_BODWakeup</b>	Show how to wake up system form Power-down mode by brown-out detector interrupt.
<b>SYS_PLLClockOutput</b>	Change system clock to different PLL frequency and output system clock from CLKO pin.
<b>TIMER_CaptureCounter</b>	Show how to use the timer2 capture function to capture timer2 counter value.
<b>TIMER_EventCounter</b>	Implement timer1 event counter function to count the external input event.
<b>TIMER_PeriodicINT</b>	Implement timer counting in periodic mode.
<b>TIMER_TimeoutWakeup</b>	Use timer0 periodic time-out interrupt event to wake up system.
<b>UART_AutoBaudRate_Master</b>	Show how to use auto baud rate detection function. This sample code needs to work with UART_AutoBaudRate_Slave.
<b>UART_AutoBaudRate_Slave</b>	Show how to use auto baud rate detection function. This sample code needs to work with UART_AutoBaudRate_Master.
<b>UART_Autoflow_Master</b>	Transmit and receive data with auto flow control. This sample code needs to work with UART_Autoflow_Slave.
<b>UART_Autoflow_Slave</b>	Transmit and receive data with auto flow control. This sample code needs to work with UART_Autoflow_Master.
<b>UART_IrDA_Master</b>	Transmit and receive data in UART IrDA mode. This

	sample code needs to work with UART_IrDA_Slave.
<b>UART_IrDA_Slave</b>	Transmit and receive data in UART IrDA mode. This sample code needs to work with UART_IrDA_Master.
<b>UART_PDMA</b>	Transmit and receive UART data with PDMA.
<b>UART_RS485_Master</b>	Transmit and receive data in UART RS485 mode. This sample code needs to work with UART_RS485_Slave.
<b>UART_RS485_Slave</b>	Transmit and receive data in UART RS485 mode. This sample code needs to work with UART_RS485_Master.
<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>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.

## 6 \SampleCode\StdDriver

<b>CLK_ClockDetector</b>	Show the usage of clock fail detector and clock frequency monitor 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>DSP_FFT</b>	Demonstrate how to call ARM CMSIS DSP library to calculate FFT.
<b>EADC_ADINT_Trigger</b>	Use ADINT interrupt to do the ADC continuous scan conversion.
<b>EADC_PDMA_PWM_Trigger</b>	Demonstrate how to trigger EADC by PWM and transfer conversion data by PDMA.
<b>EADC_PWM_Trigger</b>	Demonstrate how to trigger ADC by PWM.
<b>EADC_ResultMonitor</b>	Monitor the conversion result of channel 2 by the digital compare function.
<b>EADC_SWTRG_Trigger</b>	Trigger ADC by writing EADC_SWTRG register.
<b>EADC_Timer_Trigger</b>	Show how to trigger ADC by timer.
<b>EBI_NOR</b>	Configure EBI interface to access MX29LV320T (NOR Flash) on EBI interface.
<b>EBI_SRAM</b>	Configure EBI interface to access BS616LV4017 (SRAM) with PDMA transfer on EBI interface.
<b>FMC_ExecInSRAM</b>	Implement a code and execute in SRAM to program embedded Flash. (Support KEIL® MDK Only.)
<b>FMC_IAP</b>	Show how to reboot to LDROM functions from APROM. This sample code set VECMAP to LDROM and reset to re-boot to LDROM.
<b>FMC_RW</b>	Show how to read/program embedded flash by ISP function.

<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>I2C_EEPROM</b>	Show how to use I <sup>2</sup> C interface to access EEPROM.
<b>I2C_GCMode_Master</b>	Show how a Master uses I <sup>2</sup> C address 0x0 to write data to Slave. This sample code needs to work with I2C_GCMode_Slave.
<b>I2C_GCMode_Slave</b>	Show a Slave how to receive data from Master in GC (General Call) mode. This sample code needs to work with I2C_GCMode_Master.
<b>I2C_Master</b>	Show a Master how to access Slave. This sample code needs to work with I2C_Slave.
<b>I2C_Slave</b>	Show how to set I <sup>2</sup> C in Slave mode and receive the data from Master. This sample code needs to work with I2C_Master.
<b>I2C_SMBus</b>	Show how to control SMBus interface and use SMBus protocol between Host and Slave.
<b>I2C_Wakeup_Master</b>	Show how to wake up MCU from Power-down. This sample code needs to work with I2C_Wakeup_Slave.
<b>I2C_Wakeup_Slave</b>	Show how to wake up MCU from Power-down mode through I <sup>2</sup> C interface. This sample code needs to work with I2C_Wakeup_Master.
<b>I2S_Master</b>	Configure SPI1 as I <sup>2</sup> S Master mode and demonstrate how I <sup>2</sup> S works in Master mode. This sample code needs to work with I2S_Slave.
<b>I2S_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 I2S_Master.
<b>PDMA</b>	Use PDMA channel 2 to transfer data from memory to memory.
<b>PDMA_Scatter_Gather</b>	Use PDMA channel 5 to transfer data from memory to memory by scatter-gather mode.
<b>PDMA_ScatterGather_PingPongBuffer</b>	Use PDMA to implement Ping-Pong buffer by scatter-gather mode (memory to memory).
<b>PWM_Capture</b>	Capture the PWM1 Channel 0 waveform by PWM1 Channel 2.
<b>PWM_DeadZone</b>	Demonstrate how to use PWM Dead Zone function.
<b>PWM_DoubleBuffer</b>	Change duty cycle and period of output waveform by PWM Double Buffer function.
<b>PWM_PDMA_Capture</b>	Capture the PWM1 Channel 0 waveform by PWM1 Channel 2, and use PDMA to transfer captured data.
<b>RTC_AlarmWakeup</b>	Use RTC alarm interrupt event to wake up system.
<b>RTC_SpareRegisterRW</b>	Show how to access RTC spare registers.
<b>RTC_TimeAndTick</b>	Get the current RTC data/time per tick.
<b>SC_ReadATR</b>	Read the smartcard ATR from smartcard 0 interface.
<b>SCUART_TxRx</b>	Show smartcard UART mode by connecting PA.0 and PA.1 pins.
<b>SPI_Loopback</b>	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.
<b>SPI_MasterMode</b>	Configure SPI0 as Master mode and demonstrate how to communicate with an off-chip SPI Slave device. Needs to work with SPI_SlaveMode.
<b>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>SPI_SlaveMode</b>	Configure SPI0 as Slave mode and demonstrate how to communicate with an off-chip SPI Master device. This sample code needs to work with SPI_MasterMode.
<b>SYS_BODWakeup</b>	Show how to wake up system form Power-down mode by brown-out detector interrupt.
<b>SYS_PLLClockOutput</b>	Change system clock to different PLL frequency and output system clock from CLKO pin.
<b>TIMER_EventCounter</b>	Implement timer1 event counter function to count the external input event.
<b>TIMER_CaptureCounter</b>	Show how to use the timer2 capture function to capture timer2 counter value.
<b>TIMER_Delay</b>	Show how to use timer0 to create various delay time.
<b>TIMER_PeriodicINT</b>	Implement timer counting in periodic mode.
<b>TIMER_TimeoutWakeup</b>	Use timer0 periodic time-out interrupt event to wake up system.
<b>UART_AutoBaudRate_Master</b>	Show how to use auto baud rate detection function. This sample code needs to work with UART_AutoBaudRate_Slave.
<b>UART_AutoBaudRate_Slave</b>	Show how to use auto baud rate detection function. This sample code needs to work with UART_AutoBaudRate_Master.
<b>UART_Autoflow_Master</b>	Transmit and receive data with auto flow control. This sample code needs to work with UART_Autoflow_Slave.
<b>UART_Autoflow_Slave</b>	Transmit and receive data with auto flow control. This sample code needs to work with UART_Autoflow_Master.
<b>UART_IrDA_Master</b>	Transmit and receive data in UART IrDA mode. This sample code needs to work with UART_IrDA_Slave.
<b>UART_IrDA_Slave</b>	Transmit and receive data in UART IrDA mode. This sample code needs to work with UART_IrDA_Master.

<b>UART_PDMA</b>	Transmit and receive UART data with PDMA.
<b>UART_RS485_Master</b>	Transmit and receive data in UART RS485 mode. This sample code needs to work with UART_RS485_Slave.
<b>UART_RS485_Slave</b>	Transmit and receive data in UART RS485 mode. This sample code needs to work with UART_RS485_Master.
<b>UART_TxRxFunction</b>	Transmit and receive data from PC terminal through RS232 interface.
<b>UART_Wakeup</b>	Show how to wake up system form Power-down mode by UART interrupt.
<b>USBD_Audio_HID_NAU8822</b>	Implement a USB audio class device with HID key. NAU8822 is used in this sample code to play the audio data from Host. It also supports to record data from NAU8822 to Host.
<b>USBD_Audio_NAU8822</b>	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.
<b>USBD_HID_Keyboard</b>	Show how to implement a USB keyboard device. This sample code supports to use GPIO to simulate key input.
<b>USBD_HID_Mouse</b>	Show how to implement a USB mouse device. The mouse cursor will move automatically when this mouse device connecting to PC by USB.
<b>USBD_HID_MouseKeyboard</b>	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.
<b>USBD_HID_Transfer</b>	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.
<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 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_MassStorage_CDROM</b>	Demonstrate how to simulate a USB CD-ROM device.
<b>USBD_MassStorage_DataFlash</b>	Use embedded data flash as storage to implement a USB Mass-Storage device.
<b>USBD_Micro_Printer</b>	Show 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>	Implement a USB composite device with virtual COM port and keyboard functions.
<b>USBD_VCOM_and_HID_Transfer</b>	Demonstrate how to implement a composite device (VCOM and HID Transfer). It supports one virtual COM port and transfers 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_MassStorage</b>	Implement a USB composite device. It supports one virtual COM port and one USB Mass-Storage device.
<b>USBD_VCOM_DualPort</b>	Demonstrate how to implement a USB dual virtual COM port device.
<b>USBD_VCOM_SinglePort</b>	Implement a USB virtual COM port device. It supports one virtual COM port.
<b>USBH_AOA</b>	An Android Open Accessory (AOA) device sample.
<b>USBH_Audio_Class</b>	Show how to implement a USB Host and recognize a complex of audio (speaker, microphone) device.



<b>USBH_HID</b>	Show how to implement a USB Host and recognize a HID device when device plug-in.
<b>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>USBH_HID_MultiDevice</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>	Show how to implement a USB Host with a file system to read/write a file on USB Mass Storage.
<b>USBH_UMAS_FileRW</b>	Show how to implement a USB Host with a file system to read/write a file on USB Mass Storage.
<b>WDT_TimeoutWakeup AndReset</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.*