# NUC029xEE CMSIS BSP Directory

Directory Introduction for 32-bit NuMicro™ Family

## Directory Information

| | |
|---|---|
| **Document** | Driver reference manual and revision history. |
| **Library** | Driver header and source files. |
| **SampleCode** | Driver sample code. |

*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

# TABLE OF CONTENTS

# 1 Document Information

| | |
|---|---|
| **Revision History** | Show all the revision history about specific BSP. |
| **NuMicro NUC029xEE Driver Reference Guide.chm** | The usage of drivers in NUC029xEE Series BSP. |

# 2 Library Information

| | |
|---|---|
| **CMSIS** | CMSIS definitions by ARM® Corp. |
| **Device** | CMSIS compliant device header file. |
| **StdDriver** | All peripheral driver header and source files. |

# 3 Sample Code Information

| | |
|---|---|
| **Hard_Fault_Sample** | Show hard fault information when hard fault happened. |
| **ISP** | Sample codes for In-System-Programming. |
| **Template** | Software Development Template. |
| **Semihost** | A sample code to show how to debug with semihost message print. |
| **RegBased** | The sample codes which access control registers directly. |
| **StdDriver** | NUC029xEE Series Driver Samples |

# 4 SampleCode\ISP

| | |
|---|---|
| **ISP_DFU** | In-System-Programming Sample code through USB interface and following Device Firmware Upgrade Class Specification. |
| **ISP_HID** | In-System-Programming Sample code through USB HID interface. |
| **ISP_I2C** | In-System-Programming Sample code through I2C interface. |
| **ISP_RS485** | In-System-Programming Sample code through RS485 interface. |
| **ISP_SPI** | In-System-Programming Sample code through SPI interface. |
| **ISP_UART** | In-System-Programming Sample code through UART interface. |

# 5 SampleCode\RegBased

## System Manager (SYS)

| | |
|---|---|
| **SYS** | Demonstrate how to change system clock to different PLL frequency and output system clock from CLKO pin. |
| **SYS_PowerDown_MinCurrent** | Demonstrate how to minimize power consumption when entering power down mode |

## Flash Memory Controller (FMC)

| | |
|---|---|
| **FMC_RW** | Demonstrate how to read/program embedded flash by ISP function. |

## General Purpose I/O (GPIO)

| | |
|---|---|
| **GPIO_EINTAndDebounce** | Demonstrate how to use GPIO external interrupt function and de-bounce function. |
| **GPIO_INT** | Demonstrate how to use GPIO interrupt function. |
| **GPIO_OutputInput** | Demonstrate how to set GPIO pin mode and use pin data input/output control. |
| **GPIO_PowerDown** | Demonstrate how to wake-up form Power-down mode by GPIO interrupt. |

## PDMA Controller (PDMA)

| | |
|---|---|
| **PDMA** | Demonstrate how to use PDMA channel 6 to transfer data from memory to memory. |

## Timer Controller (TIMER)

| | |
|---|---|
| **TIMER_Capture** | Demonstrate how to use timer2 capture event to capture timer2 counter value. |

| | |
|---|---|
| **TIMER_Counter** | Demonstrate how to use timer1 counter input function to count the input event. |
| **TIMER_PeriodicINT** | Demonstrate how to perform timer counting in periodic mode. |
| **TIMER_PowerDown** | Demonstrate how to use timer0 toggle-output interrupt event to wake up system. |

## Watchdog Timer (WDT)

| | |
|---|---|
| **WDT_PowerDown** | Demonstrate how to use WDT time-out interrupt event to wake-up system. |
| **WDT_TimeoutINT** | Select one WDT time-out interval period time to generate time-out interrupt event. |
| **WDT_TimeoutReset** | Demonstrate how to cause WDT time-out reset system event while WDT time-out reset delay period expired. |

## Window Watchdog Timer (WWDT)

| | |
|---|---|
| **WWDT_CompareINT** | Select one WWDT window compare value to generate window compare match interrupt event. |

## Real Timer Clock (RTC)

| | |
|---|---|
| **RTC_PowerDown** | Demonstrate how to use RTC alarm interrupt event to wake-up system. |
| **RTC_TimeAndTick** | Demonstrate how to get the current RTC data/time per tick. |

## PWM Generator and Capture Timer (PWM)

| | |
|---|---|
| **PWM_Capture** | Demonstrate how to use PWMB Channel 2 captures PWMB Channel 1 Waveform. |
| **PWM_DeadZone** | Demonstrate how to use PWM Dead Zone function. |

| | |
|---|---|
| **PWM_DoubleBuffer** | Use PWM Double Buffer function to change duty cycle and period of output waveform. |

## UART Interface Controller (UART)

| | |
|---|---|
| **UART_Autoflow_Master** | Demonstrate how to transmit and receive data with auto flow control. The sample code needs to work with UART_Autoflow_Slave. |
| **UART_Autoflow_Slave** | Demonstrate how to transmit and receive data with auto flow control. The sample code needs to work with UART_Autoflow_Master. |
| **UART_IrDA_Master** | Demonstrate how to transmit and receive data in UART IrDA mode. The sample code needs to work with UART_IrDA_Slave. |
| **UART_IrDA_Slave** | Demonstrate how to transmit and receive data in UART IrDA mode. The sample code needs to work with UART_IrDA_Master. |
| **UART_LIN** | Demonstrate how to transmit LIN header and response. |
| **UART_PDMA** | Transmit and receive UART data with PDMA. |
| **UART_RS485_Master** | Demonstrate how to transmit and receive data in UART RS485 mode. The sample code needs to work with UART_RS485_Slave. |
| **UART_RS485_Slave** | Demonstrate how to transmit and receive data in UART RS485 mode. The sample code needs to work with UART_RS485_Master. |
| **UART_TxRx_Function** | Demonstrate how UART transmit and receive data from PC terminal through RS232 interface. |
| **UART_Wakeup** | Show how to wake up system form Power-down mode by UART interrupt. |

## Serial Peripheral Interface (SPI)

| | |
|---|---|
| **SPI_Loopback** | Implement SPI Master loop back transfer. This sample code needs to connect SPI0_MISO0 pin and SPI0_MOSI0 pin together. It will compare the received data with transmitted data. |
| **SPI_MasterFifoMode** | 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_Loopback** | Demonstrate SPI data transfer with PDMA. SPI0 will be configured as Master mode and SPI1 will be configured as Slave mode. Both TX PDMA function and RX PDMA function will be enabled. |
| **SPI_SlaveFifoMode** | 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. |

## I²C Serial Interface Controller (I²C)

| | |
|---|---|
| **I2C_EEPROM** | Demonstrate how to access EEPROM by I²C interface. |
| **I2C_GCMode_Master** | Demonstrate how a Master uses I²C address 0x0 to write data to I²C Slave. Needs to work with I2C_GCMode_SLAVE sample code. |
| **I2C_GCMode_Slave** | Demonstrate how to receive Master data in GC (General Call) mode. Needs to work with I2C_GCMode_MASTER sample code. |
| **I2C_Master** | Demonstrate how a Master access Slave. Needs to work with I2C_SLAVE sample code. |
| **I2C_Slave** | Demonstrate how to set I²C in slave mode to receive the data of a Master. Needs to work with I2C_MASTER sample code. |
| **I2C_Wakeup_Master** | Demonstrate how to set I2C to wake up MCU from Power-down mode. This sample code needs to work with I2C_Wakeup_Slave. |

| | |
|---|---|
| **I2C_Wakeup_Slave** | Demonstrate how to set I2C to wake up MCU from Power-down mode. This sample code needs to work with I2C_Wakeup_Master. |

## External Bus Interface (EBI)

| | |
|---|---|
| **EBI_NOR** | Configure EBI interface to access W39L040P (NOR Flash) on the EBI interface. |
| **EBI_SRAM** | Configure EBI interface to access BS616LV4017 (SRAM) with PDMA transfer on the EBI interface. |

## CRC Controller (CRC)

| | |
|---|---|
| **CRC_CCITT** | Perform CRC_CCITT operation and get the CRC checksum result. |
| **CRC_8** | Perform CRC-8 operation and get the CRC checksum result. |

## Analog-to-Digital Converter (ADC)

| | |
|---|---|
| **ADC_ContinuousScanMode** | Perform A/D Conversion with ADC continuous scan mode. |
| **ADC_MeasureAVDD** | Measure AVDD voltage by ADC. |
| **ADC_PwmTrigger** | Demonstrate how to trigger ADC by PWM. |
| **ADC_ResultMonitor** | Monitor the conversion result of Channel 2 by the digital compare function. |
| **ADC_SingleCycleScanMode** | Perform A/D Conversion with ADC single cycle scan mode. |
| **ADC_SingleMode** | Perform A/D Conversion with ADC single mode. |

# 6 SampleCode\StdDriver

## System Manager (SYS)

| | |
|---|---|
| **SYS** | Demonstrate how to change system clock to different PLL frequency and output system clock from CLKO pin. |
| **SYS_PowerDown_MinCurrent** | Demonstrate how to minimize power consumption when entering power down mode |

## Flash Memory Controller (FMC)

| | |
|---|---|
| **FMC_IAP** | Show how to call LDROM functions from APROM. The code in APROM will look up the table at 0x100E00 to get the address of function of LDROM and call the function. |
| **FMC_RW** | Demonstrate how to read/program embedded flash by ISP function. |

## General Purpose I/O (GPIO)

| | |
|---|---|
| **GPIO_EINTAndDebounce** | Demonstrate how to use GPIO external interrupt function and de-bounce function. |
| **GPIO_INT** | Demonstrate how to use GPIO interrupt function. |
| **GPIO_OutputInput** | Demonstrate how to set GPIO pin mode and use pin data input/output control. |
| **GPIO_PowerDown** | Demonstrate how to wake-up form Power-down mode by GPIO interrupt. |

## PDMA Controller (PDMA)

| | |
|---|---|
| **PDMA** | Demonstrate how to use PDMA channel 6 to transfer data from memory to memory. |

## Timer Controller (TIMER)

| | |
|---|---|
| **TIMER_Capture** | Demonstrate how to use timer2 capture event to capture timer2 counter value. |
| **TIMER_Counter** | Demonstrate how to use timer1 counter input function to count the input event. |
| **TIMER_PeriodicINT** | Demonstrate how to perform timer counting in periodic mode. |
| **TIMER_PowerDown** | Demonstrate how to use timer0 toggle-output interrupt event to wake-up system. |

## Watchdog Timer (WDT)

| | |
|---|---|
| **WDT_PowerDown** | Demonstrate how to use WDT time-out interrupt event to wake-up system. |
| **WDT_TimeoutINT** | Select one WDT time-out interval period time to generate time-out interrupt event. |
| **WDT_TimeoutReset** | Demonstrate how to cause WDT time-out reset system event while WDT time-out reset delay period expired. |

## Window Watchdog Timer (WWDT)

| | |
|---|---|
| **WWDT_CompareINT** | Select one WWDT window compare value to generate window compare match interrupt event. |

## Real Timer Clock (RTC)

| | |
|---|---|
| **RTC_PowerDown** | Demonstrate how to use RTC alarm interrupt event to wake-up system. |
| **RTC_TimeAndTick** | Demonstrate how to get the current RTC data/time per tick. |

## PWM Generator and Capture Timer (PWM)

| | |
|---|---|
| **PWM_Capture** | Demonstrate how to use PWMB Channel 2 captures PWMB Channel 1 Waveform. |
| **PWM_DeadZone** | Demonstrate how to use PWM Dead Zone function. |
| **PWM_DoubleBuffer** | Use PWM Double Buffer function to change duty cycle and period of output waveform. |

## UART Interface Controller (UART)

| | |
|---|---|
| **UART_Autoflow_Master** | Demonstrate how to transmit and receive data with auto flow control. The sample code needs to work with UART_Autoflow_Slave. |
| **UART_Autoflow_Slave** | Demonstrate how to transmit and receive data with auto flow control. The sample code needs to work with UART_Autoflow_Master. |
| **UART_IrDA_Master** | Demonstrate how to transmit and receive data in UART IrDA mode. The sample code needs to work with UART_IrDA_Slave. |
| **UART_IrDA_Slave** | Demonstrate how to transmit and receive data in UART IrDA mode. The sample code needs to work with UART_IrDA_Master. |
| **UART_LIN** | Demonstrate how to transmit LIN header and response. |
| **UART_PDMA** | Transmit and receive UART data with PDMA. |
| **UART_RS485_Master** | Demonstrate how to transmit and receive data in UART RS485 mode. The sample code needs to work with UART_RS485_Slave. |
| **UART_RS485_Slave** | Demonstrate how to transmit and receive data in UART RS485 mode. The sample code needs to work with UART_RS485_Master. |
| **UART_TxRx_Function** | Demonstrate how UART transmit and receive data from PC terminal through RS232 interface. |

| | |
|---|---|
| **UART_Wakeup** | Show how to wake up system form Power-down mode by UART interrupt. |

## USB Device Controller (USBD)

| | |
|---|---|
| **USBD_Audio_NAU8822** | Demonstrate how to implement a USB audio class device. NAU8822 is used in this sample code to play the audio data from Host. It also supports to record data from NAU8822 to Host. |
| **USBD_HID_Keyboard** | Demonstrate how to implement a USB keyboard device. It supports to use GPIO to simulate key input. |
| **USBD_HID_Mouse** | Demonstrate 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** | Demonstrate how to implement a USB mouse function and a USB keyboard on the same USB device. The mouse cursor will move automatically when this mouse device connecting to PC. This sample code uses a GPIO to simulate key input. |
| **USBD_HID_Transfer** | Transfer data between USB device and PC through USB HID interface. A windows tool is also included in this sample code to connect with USB device. |
| **USBD_HID_Transfer_and_Keyboard** | Demonstrate how to implement a composite device. (HID Transfer and keyboard) Transfer data between USB device and PC through USB HID interface. A windows tool is also included in this sample code to connect with USB device. |
| **USBD_HID_Transfer_and_MSC** | Demonstrate how to implement a composite device. (HID Transfer and Mass storage) Transfer data between USB device and PC through USB HID interface. A windows tool is also included in this sample code to connect with a USB device. |
| **USBD_MassStorage_CDROM** | Demonstrate how to simulate a USB CD-ROM device. |
| **USBD_MassStorage_DataFlash** | Demonstrate how to implement a USB Mass-Storage. |

| | |
|---|---|
| | It uses embedded data flash as storage. |
| **USBD_MassStorage_SDCard** | Implement a SD card reader. |
| **USBD_Micro_Printer** | Show how to implement a USB micro printer device. |
| **USBD_Printer_and_HID_Transfer** | Demonstrate how to implement a composite device (USB micro printer device and HID Transfer). Transfer data between a USB device and PC through a USB HID interface. A windows tool is also included in this sample code to connect with a USB device. |
| **USBD_VCOM_and_HID_Keyboard** | Implement a USB composite device with virtual COM port and keyboard functions. |
| **USBD_VCOM_and_HID_Transfer** | Demonstrate how to implement a composite device (VCOM and HID Transfer). It supports one virtual COM port and transfers data between a USB device and PC through a USB HID interface. A windows tool is also included in this sample code to connect with a USB device. |
| **USBD_VCOM_and_MassStorage** | Implement a USB composite device. It supports one virtual COM port and one USB Mass-Storage device. |
| **USBD_VCOM_DualPort** | Demonstrate how to implement a USB dual virtual COM port device. |
| **USBD_VCOM_SinglePort** | Implement a USB virtual COM port device. It supports one virtual COM port. |

## Serial Peripheral Interface (SPI)

| | |
|---|---|
| **SPI_Loopback** | Implement SPI Master loop back transfer. This sample code needs to connect SPI0_MISO0 pin and SPI0_MOSI0 pin together. It will compare the received data with transmitted data. |
| **SPI_MasterFIFOMode** | Configure SPI0 as Master mode and demonstrate how to communicate with an off-chip SPI Slave device with FIFO mode. This sample code needs to work with SPI_SlaveFifoMode sample code. |

| | |
|---|---|
| **SPI_PDMA_Loopback** | Demonstrate SPI data transfer with PDMA. SPI0 will be configured as Master mode and SPI1 will be configured as Slave mode. Both TX PDMA function and RX PDMA function will be enabled. |
| **SPI_SlaveFIFOMode** | Configure SPI0 as Slave mode and demonstrate how to communicate with an off-chip SPI Master device with FIFO mode. This sample code needs to work with SPI_MasterFifoMode sample code. |

## I²C Serial Interface Controller (I²C)

| | |
|---|---|
| **I2C_EEPROM** | Demonstrate how to access EEPROM by I²C interface. |
| **I2C_GCMode_Master** | Demonstrate how a Master uses I²C address 0x0 to write data to I²C Slave. Needs to work with I2C_GCMode_SLAVE sample code. |
| **I2C_GCMode_Slave** | Demonstrate how to receive Master data in GC (General Call) mode. Needs to work with I2C_GCMode_MASTER sample code. |
| **I2C_Master** | Demonstrate how a Master access Slave. Needs to work with I2C_SLAVE sample code. |
| **I2C_Slave** | Demonstrate how to set I²C in slave mode to receive the data of a Master. Needs to work with I2C_MASTER sample code. |
| **I2C_Wakeup_Master** | Demonstrate how to wake-up MCU from power-down. Needs to work with I2C_Wakeup_Slave sample code. |
| **I2C_Wakeup_Slave** | Demonstrate how to set I²C to wake-up MCU from power-down mode. Needs to work with I2C_Wakeup_Master sample code. |

## External Bus Interface (EBI)

| | |
|---|---|
| **EBI_NOR** | Configure EBI interface to access W39L040P (NOR Flash) on EBI interface. |

| | |
|---|---|
| **EBI_SRAM** | Configure EBI interface to access BS616LV4017 (SRAM) with PDMA transfer on EBI interface. |

## CRC Controller (CRC)

| | |
|---|---|
| **CRC_8** | Perform CRC-8 operation and get the CRC checksum result. |
| **CRC_CCITT** | Perform CRC_CCITT operation and get the CRC checksum result. |

## Analog-to-Digital Converter (ADC)

| | |
|---|---|
| **ADC_ContinuousScanMode** | Demonstrate how to use continuous scan mode and finishes two cycles of conversion for the specified channels. |
| **ADC_MeasureAVDD** | Measure AVDD voltage by ADC. |
| **ADC_PwmTrigger** | Demonstrate how to trigger ADC by PWM. |
| **ADC_ResultMonitor** | Demonstrate how to use the digital compare function to monitor the conversion result of channel 2. |
| **ADC_SingleCycleScanMode** | Demonstrate how to use single cycle scan mode and finishes one cycle of conversion for the specified channels. |
| **ADC_SingleMode** | Demonstrate how to use single mode and finishes the conversion of the specified channel. |

## 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.*