

M2354 Series CMSIS BSP Guide

Directory Introduction for 32-bit NuMicro™ Family

Directory Information

Please extract the “M2354_Series_BSP_CMSIS_V3.00.003.zip” file firstly, and then put the “M2354_Series_BSP_CMSIS_V3.00.003” folder into the working folder (e.g. .\Nuvoton\BSP Library\).

To experience the powerful features of M2354 in few minutes, please refer to NuMaker-PFM-M2354 Board Quick Start Guide. You can select the sample code of your interest to download and execute on the M2354 board. For example, you can try the TrustZone® technology in M2354 that provides system-wide hardware isolation for trusted software with the sample code in the folder BSP\SampleCode\TrustZone. This folder includes the sample code of TrustZone® for collaborative secure software development, Hard Fault handling and a TrustZone® template. You can open the project files to build them with Keil® MDK, IAR or Eclipse, and then download and trace them on the M2354 board to see how the TrustZone® works.

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. |

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 | 4 |
| 2 | LIBRARY | 5 |
| 3 | SAMPLE CODE | 6 |
| 4 | THIRDPARTY | 8 |
| 5 | SAMPLECODE\ATTACKDETECTION | 9 |
| 6 | SAMPLECODE\CARDREADER | 10 |
| 7 | SAMPLECODE\CORTEXM23 | 11 |
| 8 | SAMPLECODE\CRYPTO | 12 |
| 9 | SAMPLECODE\ISP | 13 |
| 10 | SAMPLECODE\NUMAKER | 14 |
| 11 | SAMPLECODE\NUMAKERIOT | 15 |
| 12 | SAMPLECODE\POWERMANAGEMENT | 16 |
| 13 | SAMPLECODE\SECUREAPPLICATION | 17 |
| 14 | SAMPLECODE\STDDRIVER | 18 |
| | System Manager (SYS) | 18 |
| | Clock Controller (CLK) | 18 |
| | Flash Memory Controller (FMC) | 18 |
| | General Purpose I/O (GPIO) | 19 |
| | PDMA Controller (PDMA) | 19 |
| | Timer Controller (TIMER) | 20 |
| | Watchdog Timer (WDT) | 21 |
| | Extra Watchdog Timer (EWDT) | 21 |
| | Window Watchdog Timer (WWDT) | 21 |
| | Extra Window Watchdog Timer (EWWDT) | 21 |

| | |
|--|-----------|
| Real Timer Clock (RTC) | 21 |
| Basic PWM Generator and Capture Timer (BPWM)..... | 22 |
| Enhance PWM Generator and Capture Timer (EPWM) | 22 |
| UART Interface Controller (UART)..... | 23 |
| Smart Card Host Interface (SC)..... | 23 |
| I ² S Controller (I ² S)..... | 24 |
| Serial Peripheral Interface (SPI)..... | 24 |
| Quadrature Encoder Interface (QEI)..... | 25 |
| Quad Serial Peripheral Interface (QSPI) | 25 |
| I ² C Serial Interface Controller (I ² C)..... | 25 |
| Universal Serial Control Interface Controller - UART Mode (USCI-UART) | 26 |
| Universal Serial Control Interface Controller - SPI Mode (USCI-SPI)..... | 26 |
| Universal Serial Control Interface Controller - I ² C Mode (USCI-I2C) | 27 |
| External Bus Interface (EBI) | 28 |
| USB 1.1 Device Controller (USBD) | 28 |
| USB 1.1 Host Controller (USBH) | 30 |
| CRC Controller (CRC) | 31 |
| Enhance 12-bit Analog-to-Digital Converter (EADC) | 31 |
| Digital-to-Analog Converter (DAC) | 32 |
| Analog Comparator Controller (ACMP) | 32 |
| Controller Area Network (CAN) | 32 |
| Liquid Crystal Display (LCD) | 33 |
| Key Store (KS)..... | 33 |
| Cryptographic Accelerator (CRYPTO) | 33 |
| USB On-The-Go (OTG)..... | 35 |
| Debug Protection Mechanism (DPM) | 35 |
| Enhanced Input Capture Timer (ECAP) | 35 |
| Tamper Controller (TC)..... | 35 |
| Random Number Generator (RNG)..... | 35 |
| Secure Digital Host Controller (SDH)..... | 35 |
| 15 SAMPLECODE\TRUSTZONE | 36 |
| 16 SAMPLECODE\XOM | 37 |

1 Document

| | |
|--|---|
| CMSIS.html | <p>Introduction of CMSIS version 5. CMSIS components included CMSIS-CORE, CMSIS-Driver, CMSIS-DSP, etc.</p> <ul style="list-style-type: none"> ● 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). |
| Revision History.pdf | The revision history of M2354 Series BSP. |
| NuMicro M2354 Series Driver Reference Guide.chm | The usage of drivers in M2354 Series BSP. |

2 Library

| | |
|---------------------|--|
| CMSIS | Cortex® Microcontroller Software Interface Standard (CMSIS) V5 definitions by ARM® Corp. |
| Device | CMSIS compliant device header file. |
| LCDLib | Library for controlling LCD module. |
| NuMaker | Specific libraries for M2354 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 |
| CortexM23 | Cortex®-M23 sample code. |
| Crypto | Crypto sample code using MbedTLS library. |
| FreeRTOS | Simple FreeRTOS™ demo code. |
| Hard_Fault_Sample | <p>Show hard fault information when hard fault happened.</p> <p>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.</p> <p>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.</p> |
| ISP | Sample codes for In-System-Programming. |
| NuMaker | Sample codes for NuMaker-PFM-M2354 board. |
| NuMakerIoT | Sample codes for NuMaker-IoT-M2354 board. |
| PowerManagement | Power management sample code. |
| SecureApplication | Sample codes for secure application |
| Semihost | Show how to print and get character through IDE console window. |

| | |
|------------------|---|
| StdDriver | Demonstrate the usage of M2354 series MCU peripheral driver APIs. |
| TrustZone | Includes the demo of secure codes and non-secure codes. |
| XOM | Demonstrate how to create XOM library and use it. |

4 ThirdParty

| | |
|-----------------|---|
| BME680 | Sensor driver for BME680 gas sensor. |
| FatFs | An open source FAT/exFAT file system library. |
| FreeRTOS | FreeRTOS porting for M2354. |
| MBEDTLS | An open source crypto library. |
| SX1276 | The Semtech SX1276 LoRa transceiver 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. |
| TAMPER_FaultInjectionDetection | Show the usage of TAMPER power glitch positive and negative detection function. |
| TAMPER_FrequencyDetection | Show the usage of core HXT detection and LXT clock frequency monitor function. |
| TAMPER_VoltageDetection | Show the usage of TAMPER voltage detection function. |

6 SampleCode\CardReader

| | |
|-----------|--|
| USBD_CCID | USB CCID Smartcard Reader sample code. |
|-----------|--|

7 SampleCode\CortexM23

| | |
|-----|---|
| MPU | Demonstrate the usage of Cortex®-M23 MPU. |
|-----|---|

8 SampleCode\Crypto

| | |
|-----------------------|---|
| mbedTLS_AES | Show how mbedTLS AES function works. |
| mbedTLS_ECDH | Show how mbedTLS ECDH function works. |
| mbedTLS_ECDSA | Show how mbedTLS ECDSA function works. |
| mbedTLS_RSA | Show how mbedTLS RSA function works. |
| mbedTLS_SHA256 | Show how mbedTLS SHA256 function works. |

9 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 I ² C 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. |

10 SampleCode\NuMaker

| | |
|--------------------|---|
| Audio | Demonstrate how to play/record audio. NAU88L25 is used in this sample code to play and record the audio data. |
| Blinky | A simple LED toggle sample code. It could be used as the startup of M2354 NuMaker board. |
| DebugUART | A simple debug message print demo. |
| Key | A simple key demo for NuMaker board. |
| RTX_Demo | Show the usage of clock fail detector and clock frequency monitor function. |
| WiFi_bypass | By pass ESP8266 to UART port. |
| Xmodem | Demonstrate how to transfer data with UART Xmodem. |

11 SampleCode\NuMakerIoT

| | |
|------------------------|--|
| BME680_and_LCD | Show the BME680 sensor data on RHE6616TP01(8-COM, 40-SEG, 1/4 Bias) LCD. |
| LoRa_Master | A simple LoRa master demo for NuMakerIoT board. |
| LoRa_Slave | A simple LoRa slave demo for NuMakerIoT board. |
| mbedOS6_AWS_IoT | mbedOS6 for AWS IoT Cloud connection. |
| WiFi_bypass | A simple WiFi demo for NuMakerIoT board. |

12 SampleCode\PowerManagement

| | |
|------------------------------------|--|
| SYS_DPDMode_Wakeup | Show how to wake up system form DPD Power-down mode by Wake-up pin(PC.0), Wake-up Timer, RTC Tick, RTC Alarm or RTC Tamper 0. |
| SYS_PowerDown_MinCurrent | Demonstrate how to minimize power consumption when entering power down mode. |
| 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 GPIO pin(PC.0), Wake-up Timer, Wake-up ACMP, RTC Tick, RTC Alarm, RTC Tamper 0, BOD or LVR. |
| SYS_SPDMode_WakeupAndReturn | Show how to continue executing code after wake-up form SPD Power-down mode by SRAM data retention function. |
| SYS_SPDMode_WakeupVTOR | Show how to continue executing code after wake-up form SPD Power-down mode by VTOR function. |
| SYS_SRAMPowerMode | Show how to select SRAM power. |

13 SampleCode\SecureApplication

| | |
|-------------------------------|---|
| SecureBootDemo | Demonstrate how to generate the first booting image, NuBL2. After NuBL2 runs, NuBL2 will authenticate NuBL32 and NuBL33 then jump to execute in NuBL32. |
| SecureISPDemo | Demonstrate how to initial SecureISP mode. |
| SecureOTABankRemapDemo | Demonstrate to do secure OTA update for NuBL32 and NuBL33 firmware by NuBL2 with bank remap. |
| SecureOTADemo | Demonstrate to update NuBL32 and NuBL33 firmware securely over the air (OTA) by NuBL2. |
| USBH_SecureISP | Use M2354 as USB host to perform SecureISP. |

14 SampleCode\StdDriver

System Manager (SYS)

| | |
|---------------------------|---|
| 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 CLK0 pin. |
| SYS_TrimIRC | Demonstrate how to use LXT to trim HIRC. |

Clock Controller (CLK)

| | |
|--------------------------------|---|
| CLK_ClockDetector | Show the usage of clock fail detector and clock frequency monitor function. |
| CLK_ClockDetectorWakeup | Show how to wake up system form Power-down mode by clock fail detector interrupt. |

Flash Memory Controller (FMC)

| | |
|---------------------------------|--|
| FMC_CRC32 | Demonstrate how to use FMC CRC32 ISP Command to calculate the CRC32 checksum of APROM and LDROM. |
| FMC_DualBank | Demonstrate how dual processes work in dual bank flash architecture. |
| FMC_DualBankFwUpgrade | Implement a firmware update mechanism based on dual bank flash architecture. |
| FMC_ExecInSRAM | Implement a code and execute in SRAM to program embedded Flash (support KEIL MDK only). |
| FMC_FwUpgradeApplication | Bank remap sample code. |
| 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_MultiWordProgram | Show how to read/program embedded flash by ISP function. |
| FMC_NonSecureISP | This sample code shows how to use Non-Secure ISP in Non-Secure world. |
| 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_XOM | This sample code shows how to configure and setup an XOM region the perform XOM function. |

General Purpose I/O (GPIO)

| | |
|-----------------------------|--|
| 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. |
| GPIO_SingleCycleIO | Show GPIO single cycle IO bus performance. |

PDMA Controller (PDMA)

| | |
|--|--|
| 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). |
| PDMA_StrideRepeat | Use PDMA0 channel 0 to transfer data from memory to memory with stride and repeat. |

| | |
|---------------------|--|
| PDMA_TimeOut | Demonstrate PDMA0 channel 1 get/clear timeout flag with UART1. |
|---------------------|--|

Timer Controller (TIMER)

| | |
|------------------------------------|---|
| TIMER_ACMPTrigger | Use ACMP to trigger Timer0 counter reset mode. |
| TIMER_CaptureClockFreq | Show how to use the capture function to capture the internal clock frequency. |
| 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~Timer5 PWM. |
| TIMER_PWM_PWDAndWakeUp | Demonstrate Timer4, Timer5 PWM output waveform in power-down and wake-up functions. |

| | |
|----------------------------|---|
| 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. |

Watchdog Timer (WDT)

| | |
|----------------------------------|--|
| 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. |
|----------------------------------|--|

Extra Watchdog Timer (EWDT)

| | |
|-----------------------------------|---|
| EWDT_TimeoutWakeupAndReset | Implement EWDT time-out interrupt event to wake up system and generate time-out reset system event while EWDT time-out reset delay period expired |
|-----------------------------------|---|

Window Watchdog Timer (WWDT)

| | |
|------------------------|--|
| WWDT_CompareINT | Show how to reload the WWDT counter value. |
|------------------------|--|

Extra Window Watchdog Timer (EWWDT)

| | |
|-------------------------|---|
| EWWDT_CompareINT | Show how to reload the EWWDT counter value. |
|-------------------------|---|

Real Timer Clock (RTC)

| | |
|---------------------------|---|
| 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 spareregister 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. |
|-------------------------|---|

Basic PWM Generator and Capture Timer (BPWM)

| | |
|----------------------------|---|
| 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. |

Enhance PWM Generator and Capture Timer (EPWM)

| | |
|---|---|
| EPWM_AccumulatorINT_Tri ggerPDMA | Demonstrate EPWM accumulator interrupt trigger PDMA. |
| EPWM_AccumulatorStopMo de | Demonstrate EPWM accumulator stop mode. |
| EPWM_Brake | Demonstrate how to use EPWM brake function. |
| EPWM_Capture | Capture the EPWM1 Channel 0 waveform by EPWM1 Channel 2. |
| EPWM_DeadTime | 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. |
|-----------------------|---|

UART Interface Controller (UART)

| | |
|--------------------------|---|
| 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_SingleWire | Transmit and receive data by UART Single-Wire 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. |

Smart Card Host Interface (SC)

| | |
|-----------------------------|---|
| 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_TxRxi | Demonstrate smartcard UART mode by connecting PB.4 and PB.5 pins. |

I²S Controller (I²S)

| | |
|-----------------------|---|
| I2S_Codec | This is an I ² S demo using NAU8822/88L25 audio codec, and used to play back the input from line-in. |
| I2S_Codec_PDMA | This is an I ² S 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. |

Serial Peripheral Interface (SPI)

| | |
|---------------------------|--|
| SPI_Flash | Access SPI flash through SPI interface. |
| SPI_HalfDuplex | Demonstrate SPI half-duplex mode. SPI0 will be configured as Master mode and SPI1 will be configured as Slave mode. Both SPI0 and SPI1 will be configured as half-duplex mode. |
| 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 I ² S Master mode and demonstrate how I ² S works in Master mode. This sample code needs to work with SPII2S_Slave sample code. |
| SPII2S_PDMA_Codec | This is an I ² S demo with PDMA function connected with audio codec. |

| | |
|-------------------------------|--|
| SPII2S_PDMA_Play | This is an I ² S demo for playing data and demonstrate how I2S works with PDMA. |
| SPII2S_PDMA_PlayRecord | This is an I ² S demo for playing and recording data with PDMA function. |
| SPII2S_PDMA_Record | This is an I ² S demo for recording data and demonstrate how I2S works with PDMA. |
| SPII2S_Slave | Configure SPI0 as I ² S Slave mode and demonstrate how I ² S works in Slave mode. This sample code needs to work with SPII2S_Master sample code. |

Quadrature Encoder Interface (QEI)

| | |
|-------------------------|---|
| QEI_CompareMatch | Show the usage of QEI compare function. |
|-------------------------|---|

Quad Serial Peripheral Interface (QSPI)

| | |
|----------------------------|--|
| QSPI_DualMode_Flash | Access SPI flash using QSPI dual mode. |
| QSPI_QuadMode_Flash | Access SPI flash using QSPI quad mode. |
| QSPI_Slave3Wire | Configure QSPI0 as Slave 3 wire 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 through a I ² C interface |
| I2C_GCMode_Master | Demonstrate how a Master uses I ² C address 0x0 to write data to I ² C Slave. This sample code needs to work with I2C_GCMode_Slave sample code. |
| I2C_GCMode_Slave | Demonstrate how to receive Master data in GC (General Call) mode. This sample code needs to work with I2C_GCMode_Master sample code. |
| 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 sample code. |
| 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 sample code. |
| I2C_SingleByte_Master | Demonstrate how to use single byte API to access slave. This sample code needs to work with I2C Slave sample code. |
| I2C_Slave | Demonstrate how to set I ² C in slave mode to receive the data from a Master. This sample code needs to work with I2C Master sample code. |
| I2C_Slave_PDMA | Demonstrate how a Slave uses PDMA Rx mode receive data from a Master. |
| I2C_Wakeup_Slave | Demonstrate how to set I ² C to wake up MCU from Power-down mode. This sample code needs to work with I2C Master sample code. |

Universal Serial Control Interface Controller - UART Mode (USCI-UART)

| | |
|-------------------------------|---|
| 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. |

Universal Serial Control Interface Controller - SPI Mode (USCI-SPI)

| | |
|--------------------------|---|
| 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. |

Universal Serial Control Interface Controller - I²C Mode (USCI-I2C)

| | |
|-----------------------------------|---|
| USCI_I2C_EEPROM | Demonstrate how to access EEPROM through a USCI_I2C interface. |
| USCI_I2C_Loopback | Demonstrate how a Master access 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 USCI_I2C_Slave sample code. |
| USCI_I2C_Master_10bit | Demonstrate how a Master use 10-bit addressing access Slave. This sample code needs to work with USCI_I2C_Slave_10bit sample code. |
| USCI_I2C_MultiBytes_Master | Demonstrate how to use multi-bytes API to access slave. This sample code needs to work with USCI_I2C_Slave sample code. |
| USCI_I2C_SingleByte_Master | Demonstrate how to use single byte API to access slave. This sample code needs to work with USCI_I2C_Slave sample code. |
| USCI_I2C_Slave | Demonstrate how to set I ² C in slave mode to receive the data from a Master. This sample code needs to work with USCI_I2C_Master sample code. |

| | |
|------------------------------|---|
| USCI_I2C_Slave_10bit | Demonstrate how to set I ² C in 10-bit addressing slave mode to receive the data from a Master. This sample code needs to work with USCI_I2C_Master_10bit sample code. |
| USCI_I2C_Wakeup_Slave | Demonstrate how to set I ² C to wake up MCU from Power-down mode. This sample code needs to work USCI_I2C_Master sample code. |

External Bus Interface (EBI)

| | |
|-----------------|--|
| 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. |

USB 1.1 Device Controller (USBD)

| | |
|--------------------------------|---|
| 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 uses 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_K | Demonstrate how to implement a composite device (HID Transfer and Keyboard). Transfer data between USB device |

| | |
|--------------------------------------|--|
| Keyboard | 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_CDR_OM | Demonstrate the emulation of USB Mass Storage Device CD-ROM. |
| USBD_Mass_Storage_Flash | Use Flash as storage to implement a USB Mass-Storage device. |
| USBD_Mass_Storage_SD | Use SD card as storage to implement a USB Mass-Storage device. |
| USBD_Mass_Storage_SDCard | Use SD card as storage to implement a USB Mass-Storage device. |
| 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. |
| USBD_VCOM_SerialEmulator_DoubleBuffer | Demonstrate how to implement a USB virtual COM port device using double buffer mode. |
| USBD_VENDOR_LBK | This sample works as a proprietary Vendor LBK device. It's created for sample USBH_VENDOR_LBK of this BSP. Vendor LBK device includes Control, Bulk, Interrupt, and Isochronous in/out endpoint pairs. Each endpoint pair receive data from host via the out-endpoint and send data back to host via the in-endpoint. |

USB 1.1 Host Controller (USBH)

| | |
|--------------------------------|--|
| 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_HID_Mouse_Keyboard | Use USB Host core driver and HID driver. This sample demonstrates how to support mouse and keyboard input. |
| 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. |
| USBH_VENDOR_LBK | This sample shows how to do transfers on a known device with a vendor driver. This sample requires a M2354 USB device running sample USBD_VENDOR_LBK be connected. |

CRC Controller (CRC)

| | |
|------------------|--|
| 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. |

Enhance 12-bit Analog-to-Digital Converter (EADC)

| | |
|------------------------------|---|
| 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. |
|------------------|--|

Digital-to-Analog Converter (DAC)

| | |
|------------------------------|--|
| 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. |

Analog Comparator Controller (ACMP)

| | |
|---------------------------|--|
| ACMP_CompareDAC | Demonstrate how ACMP compare DAC output with ACMP1_P1 value. |
| ACMP_CompareVBG | Demonstrate how ACMP compare VBG output with ACMP1_P1 value. |
| ACMP_Wakeup | Show how to wake up MCU from Power-down mode by ACMP wake-up function. |
| ACMP_WindowComapre | Demonstrate the usage of ACMP window compare function. |

Controller Area Network (CAN)

| | |
|-------------------------|---|
| CAN_BasicMode_Rx | Demonstrate CAN bus receive a message with basic mode. This sample code could work with CAN_BasicMode_Tx sample code. |
| CAN_BasicMode_Tx | Demonstrate CAN bus transmit a message with basic mode. This sample code could work with CAN_BasicMode_Rx sample code. |

| | |
|-----------------------------|--|
| 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. This sample code could work with CAN_NormalMode_Tx sample code. |
| CAN_NormalMode_Tx | Demonstrate CAN bus transmit a message with normal mode. This sample code could work with CAN_NormalMode_Rx sample code. |
| 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. |

Liquid Crystal Display (LCD)

| | |
|------------------------|--|
| LCD_Blinking | Demonstrate the LCD blinking function by using RHE6616TP01(8-COM, 40-SEG, 1/4 Bias) LCD. |
| LCD_Pixel_OnOff | Show how to set pixel on and off on RHE6616TP01(8-COM, 40-SEG, 1/4 Bias) LCD. |
| LCD_Print_Text | Show how to print text on RHE6616TP01(8-COM, 40-SEG, 1/4 Bias) LCD. |

Key Store (KS)

| | |
|------------------|---|
| KS_AESKey | Demo to use the AES in Key Store. |
| KS_ECDH | Demo to use ECC ECDH with Key Store. |
| KS_ECDSA | Demo to use the ECC ECDSA with Key Store. |

Cryptographic Accelerator (CRYPTO)

| | |
|-----------------------|---|
| CRYPTO_AES | Show Crypto IP AES-128 ECB mode encrypt/decrypt function. |
| CRYPTO_AES_CCM | Demonstrate how to encrypt/decrypt data by AES CCM. |

| | |
|--|--|
| CRYPTO_AES_GCM | Demonstrate how to encrypt/decrypt data by AES GCM. |
| CRYPTO_AES_SM4 | Show Crypto SM4 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_ECC_SM2 | Show whole ECC SM2 flow. Including private key/public key/Signature generation and Signature verification. |
| 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_SHA_SM3 | Show how to calculate SM3 hash of the data. |
| CRYPTO_SHA256_Flash | Show how to calculate SHA-256 of the data in Flash. |
| CRYPTO_RSA | Show how to use Crypto RSA engine to sign and verify signatures. |
| CRYPTO_RSA_AccessKeyStore | Use Crypto RSA engine accesses key from key store to sign and verify signatures. |
| CRYPTO_RSA_CRTBypass | Show how to use Crypto RSA engine CRT/CRT bypass mode to sign two signatures. |
| CRYPTO_RSA_CRTBypass_AccessKeyStore | Use Crypto RSA engine CRT/CRT bypass mode accesses key from key store to sign and verify signatures. |

| | |
|--------------------------------------|--|
| CRYPTO_RSA_SCAP | Show how to use Crypto RSA engine SCAP mode to sign and verify signatures. |
| CRYPTO_RSA_SCAPAccessKeyStore | Use Crypto RSA engine SCAP mode accesses key from key store to sign and verify signatures. |

USB On-The-Go (OTG)

| | |
|---------------------------|---|
| 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. |
|---------------------------|---|

Debug Protection Mechanism (DPM)

| | |
|---------------------------|---|
| DPM_UpdatePassword | Demonstrate how to update and compare Secure DPM password |
|---------------------------|---|

Enhanced Input Capture Timer (ECAP)

| | |
|--------------------------|---|
| ECAP_GetInputFreq | Show how to use ECAP interface to get input frequency |
| ECAP_GetQEIFreq | Show how to use ECAP interface to get QEI frequency. |

Tamper Controller (TC)

| | |
|-------------------------------|---|
| TAMPER_TamperDetection | Show the usage of TAMPER static tamper interrupt to wake up system or clear DataFlash data. |
|-------------------------------|---|

Random Number Generator (RNG)

| | |
|-------------------|--------------------------------------|
| RNG_Random | Demo to use ECC ECDH with Key Store. |
|-------------------|--------------------------------------|

Secure Digital Host Controller (SDH)

| | |
|------------------|--|
| SDH_FATFS | Access a SD card formatted in FAT file system. |
|------------------|--|

15 SampleCode\TrustZone

| | |
|------------------|---|
| CSSD_LED | Demonstrate how to implement code for Collaborative Secure Software Development in both secure and non-secure code. |
| HardFault | Show the hard fault usages in both secure and non-secure code. |
| Template | Demonstrate the how to implement code for secure and non-secure. |

16 SampleCode\XOM

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

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.*