M480 QSPI 4-BIT Transmission

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	This example code is used for the M480 series microcontroller (MCU) to send QSPI 4 serial data bits simultaneously.
BSP Version	M480 BSP CMSIS V3.05.001
Hardware	NuMaker-M483KG V1.1

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 and microprocessor 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

1 Overview

This example code is used to configure two QSPIs to transfer serial data bits to other device. Through 2-bit transfer mode, two QSPIs transfer the four serial data bits simultaneously. This document introduces the example code of QSPI master configuration, QSPI slave configuration, 2-bit transfer mode configuration, and four serial data bits simultaneous transmission.

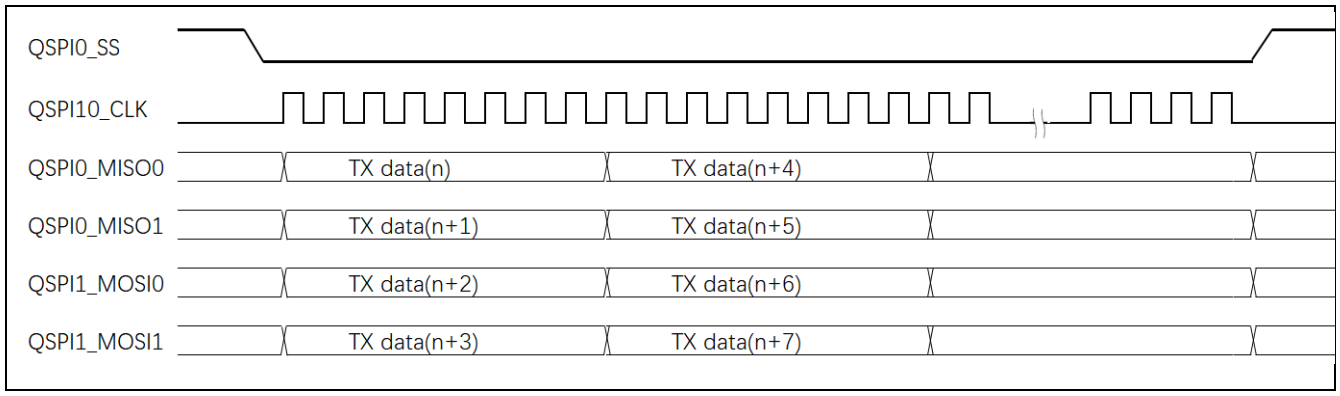

Figure 1-1 4-bit Transfer Mode Timing (Master Mode)

1.1 Demo Result

If debug UART is connected to PC when executing the example, UART tool can display debug information. The demo result is shown below.

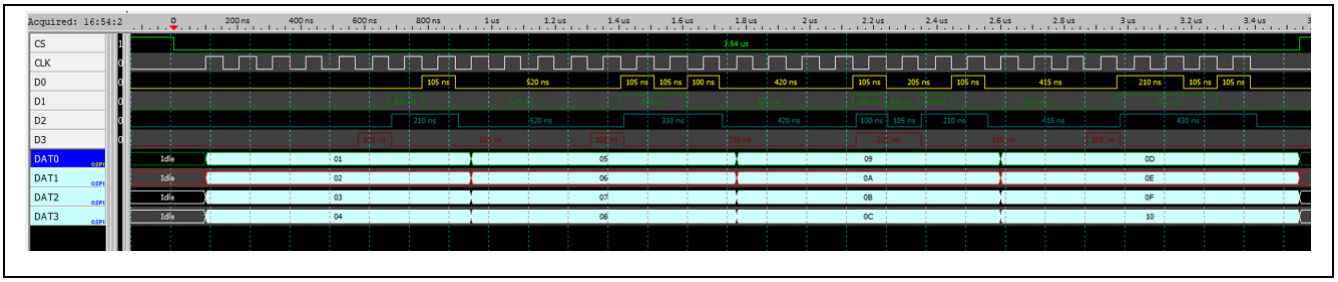

Figure 1-2 Demo Result

2 Code Description

Define TX buffer:

```
#define TEST_COUNT 16
uint8_t g_au8SourceData[TEST_COUNT];
```

QSPI clock initialization:

```
    /* Select PCLK0 as the clock source of QSPI0 */
    CLK_SetModuleClock(QSPI0_MODULE, CLK_CLKSEL2_QSPI0SEL_PCLK0, MODULE_NoMsk);
    /* Select PCLK1 as the clock source of QSPI1 */
    CLK_SetModuleClock(QSPI1_MODULE, CLK_CLKSEL3_QSPI1SEL_PCLK1, MODULE_NoMsk);

    /* Enable QSPI0 peripheral clock */
    CLK_EnableModuleClock(QSPI0_MODULE);

    /* Enable QSPI1 peripheral clock */
    CLK_EnableModuleClock(QSPI1_MODULE);
```

PIN initialization:

```
        /* Setup QSPI0 multi-function pins */
        SYS->GPA_MFPL &= ~(SYS_GPA_MFPL_PA0MFP_Msk |
                           SYS_GPA_MFPL_PA1MFP_Msk |
                           SYS_GPA_MFPL_PA2MFP_Msk |
                           SYS_GPA_MFPL_PA3MFP_Msk |
                           SYS_GPA_MFPL_PA4MFP_Msk |
                           SYS_GPA_MFPL_PA5MFP_Msk);
        SYS->GPA_MFPL |= SYS_GPA_MFPL_PA0MFP_QSPI0_MOSI0 |
                         SYS_GPA_MFPL_PA1MFP_QSPI0_MISO0 |
                         SYS_GPA_MFPL_PA2MFP_QSPI0_CLK |
                         SYS_GPA_MFPL_PA3MFP_QSPI0_SS |
                         SYS_GPA_MFPL_PA4MFP_QSPI0_MOSI1 |
                         SYS_GPA_MFPL_PA5MFP_QSPI0_MISO1;

        /* Enable QSPI0 clock pin (PA2) schmitt trigger */
        PA->SMTEN |= GPIO_SMTEN_SMTEN2_Msk;
        /* Enable QSPI0 I/O high slew rate */
        GPIO_SetSlewCtl(PA, 0x3F, GPIO_SLEWCTL_HIGH);

        /* Setup QSPI1 multi-function pins */
        SYS->GPC_MFPL &= ~(SYS_GPC_MFPL_PC2MFP_Msk |
                           SYS_GPC_MFPL_PC3MFP_Msk |
                           SYS_GPC_MFPL_PC4MFP_Msk |
```

```
                              SYS_GPC_MFPL_PC5MFP_Msk);
        SYS->GPC_MFPL |= SYS_GPC_MFPL_PC2MFP_QSPI1_MOSI0 |
                         SYS_GPC_MFPL_PC3MFP_QSPI1_MISO0 |
                         SYS_GPC_MFPL_PC4MFP_QSPI1_CLK |
                          SYS_GPC_MFPL_PC5MFP_QSPI1_SS;

        SYS->GPG_MFPH &= ~(SYS_GPG_MFPH_PG9MFP_Msk |
                           SYS_GPG_MFPH_PG10MFP_Msk);
        SYS->GPG_MFPH |= SYS_GPG_MFPH_PG9MFP_QSPI1_MISO1 |
                         SYS_GPG_MFPH_PG10MFP_QSPI1_MOSI1;
        /* Enable QSPI1 clock pin (PC4) schmitt trigger */
        PC->SMTEN |= GPIO_SMTEN_SMTEN4_Msk;
        /* Enable QSPI1 I/O high slew rate */
        GPIO_SetSlewCtl(PC, 0x3C, GPIO_SLEWCTL_HIGH);
        GPIO_SetSlewCtl(PG, 0x600, GPIO_SLEWCTL_HIGH);
```

QSPI initialization:

```
void QSPI_Init(void)
{
   /* Configure QSPI0 as a master, MSB first, 8-bit transaction, mode 0, clock is 10MHz */
    QSPI_Open(QSPI0, QSPI_MASTER, QSPI_MODE_0, 8, 10000000);
    /* Enable the automatic hardware slave select function. Select the SS pin and
       configure as low-active. */
    QSPI_EnableAutoSS(QSPI0, QSPI_SS, QSPI_SS_ACTIVE_LOW);
    /* enable QSPI0 2-bit mode */
    QSPI_ENABLE_TWOBIT_MODE(QSPI0);
    /* Set TX FIFO threshold and enable FIFO mode. */
    QSPI_SetFIFO(QSPI0, 4, 4);


    /* Configure QSPI1 as a slave, MSB first, 8-bit transaction, mode 0 */
    QSPI_Open(QSPI1, SPI_SLAVE, SPI_MODE_0, 8, (uint8_t)NULL);
    /* enable QSPI1 2-bit mode */
    QSPI_ENABLE_TWOBIT_MODE(QSPI1);
    /* Set TX FIFO threshold and enable FIFO mode. */
    QSPI_SetFIFO(QSPI1, 4, 4);
}
```

Transfer data:

```
    g_u8TxDataCount = 0;

    /*clear TX FIFO */
```

```
    QSPI_ClearTxFIFO(QSPI0);
    QSPI_ClearTxFIFO(QSPI1);

    /* send 16 byte */
    while (g_u8TxDataCount < TEST_COUNT - 2)
    {
        /* Check TX FULL flag */
        if ((QSPI_GET_TX_FIFO_FULL_FLAG(QSPI0) == 0) && (QSPI_GET_TX_FIFO_FULL_FLAG(QSPI1)
== 0))
        {
            /* Write to TX FIFO */
            /* QSPI1 slave first */
            QSPI_WRITE_TX(QSPI1, g_au8SourceData[g_u8TxDataCount + 2]); /* byte[n + 2] */
            QSPI_WRITE_TX(QSPI1, g_au8SourceData[g_u8TxDataCount + 3]); /* byte[n + 3] */

            /* QSPI0 master */
            QSPI_WRITE_TX(QSPI0, g_au8SourceData[g_u8TxDataCount]); /* byte[n] */
            QSPI_WRITE_TX(QSPI0, g_au8SourceData[g_u8TxDataCount + 1]); /*byte[n + 1] */

            g_u8TxDataCount += 4;
        }
    }

    /* Wait for transmission to finish */
    while ((SPI_IS_BUSY(QSPI0) == 1) || (SPI_IS_BUSY(QSPI0) == 1));
```

3 Software and Hardware Requirements

3.1 Software Requirements

- BSP version
  - M480 Series BSP CMSIS v3.05.001
- IDE version
  - Keil uVersion 5.26

3.2 Hardware Requirements

- Circuit components
  - NuMaker-M483 V1.1
  - PC
  - Logic analyzer

- Pin Connect

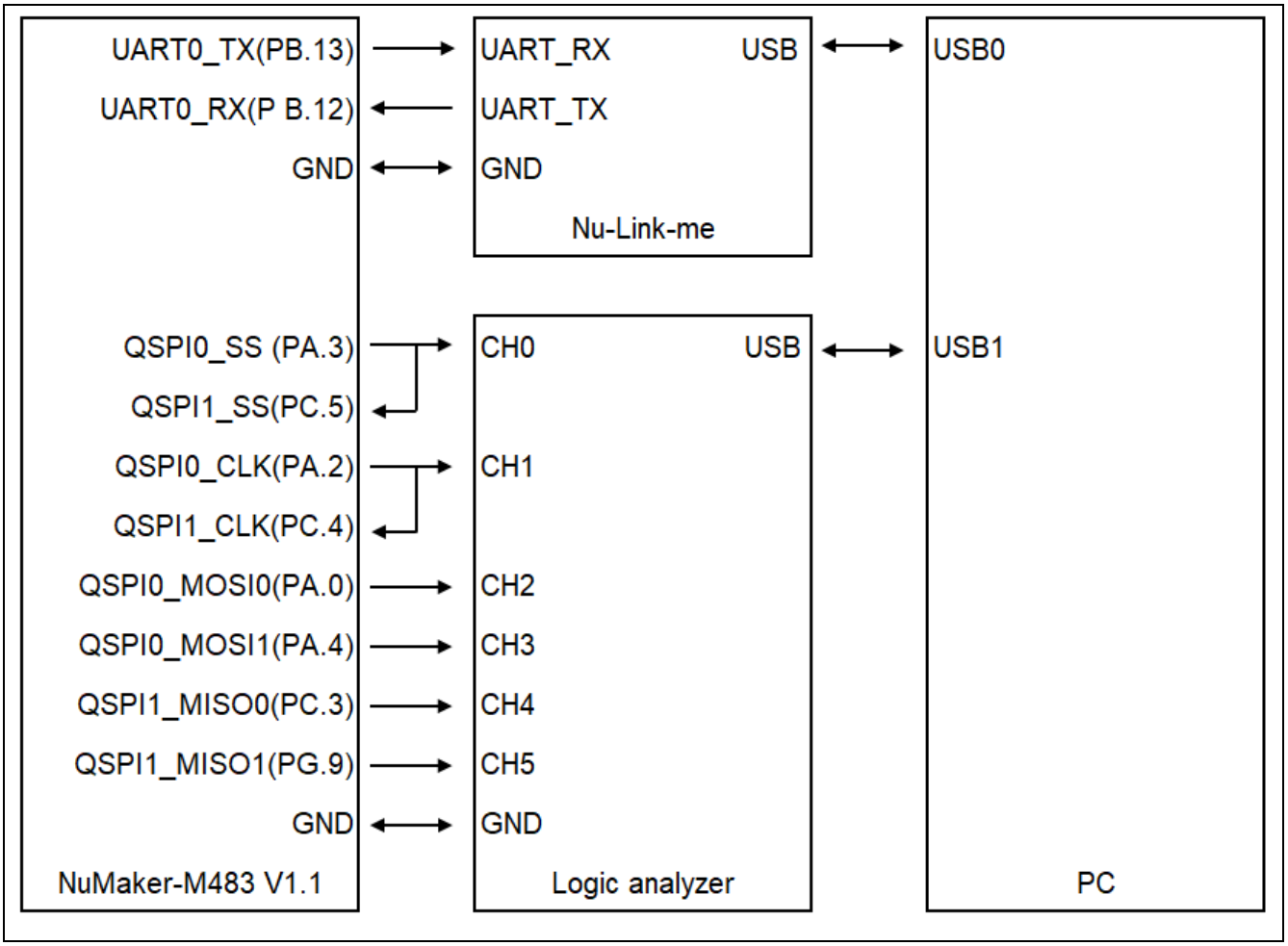

Figure 3-1 Pin Connect

4 Directory Information

The directory structure is shown below.

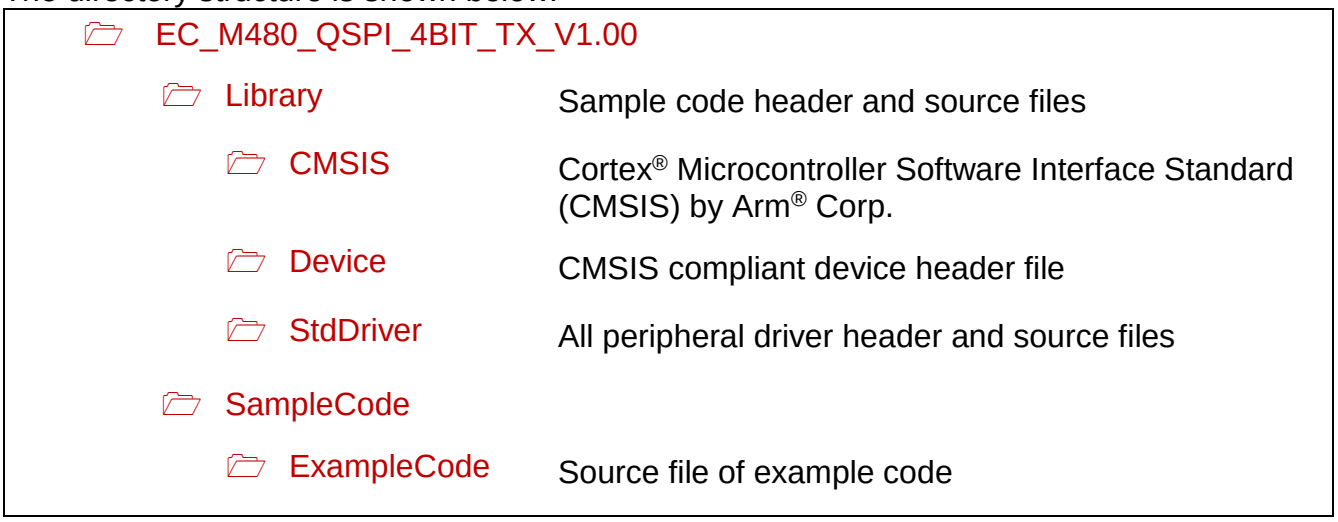

Figure 4-1 Directory Structure

5 Example Code Execution

1. Browse the sample code folder as described in the Directory Information section and double-click *M480_QSPI_4BIT_TX.uvproj*.
2. Enter Keil compile mode.
   - Build
   - Download
   - Start/Stop debug session
3. Enter debug mode.
   - Run
4. Print debug information.
   - Open *UartAssist.exe*
   - Baud rate: 115200bps
   - Press any key

6 Revision History

Date	Revision	Description
2021.12.09	1.00	1. Initially issued.

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.