M460 简单的 CANFD 发送代码

NuMicro® 32位系列微控制器范例代码介绍

文件信息

应用简述	本范例代码使用 M467 的 CANFD 接口发送报文
BSP 版本	M460_Series_BSP_CMSIS_V3.00.001
开发平台	NuMaker-M467HJ V1.0

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. 概述

使用 M467 做一个简单的、CANFD 发送报文的代码，用于 M467 CANFD 的测试，也可演示 M467 CANFD 发送报文的配置。

1.1 原理

M467 的 CANFD，每条等待发送的报文由 “2words ID + 数据字节” 组成，报文格式如图 1-1。

Bit	31			24	23			16	15		8	7		0
T0	ESI	XTD	RTR	ID[28:0]										
T1	MM[7:0]				EFC	-	FDF	BRS	DLC[3:0]	MM[15:8]		-		
T2	DB3[7:0]				DB2[7:0]					DB1[7:0]		DB0[7:0]		
T3	DB7[7:0]				DB6[7:0]					DB5[7:0]		DB4[7:0]		

图 1-1 等待发送的报文的格式

一条报文启动发送后，可能因总线忙而处于等待状态。M467 每个 CANFD 接口最多可配置 32 条等待发送报文存放在 “专用 RAM 区” 。这个专用 RAM 区有 6144Bytes，用于存放等待发送的报文、接收到的报文、及其它报文配置信息。

图 1-2 展示了与发送报文相关的寄存器：多条等待发送的报文在专用 RAM 区的首地址由寄存器 CANFDx->TXBC 配置，报文数据字节长度由寄存器 CANFDx->TXESC 配置。32 位寄存器 CANFDx->TXBRP 置 1 的位，启动对应报文发送，发送成功后自动清 0。

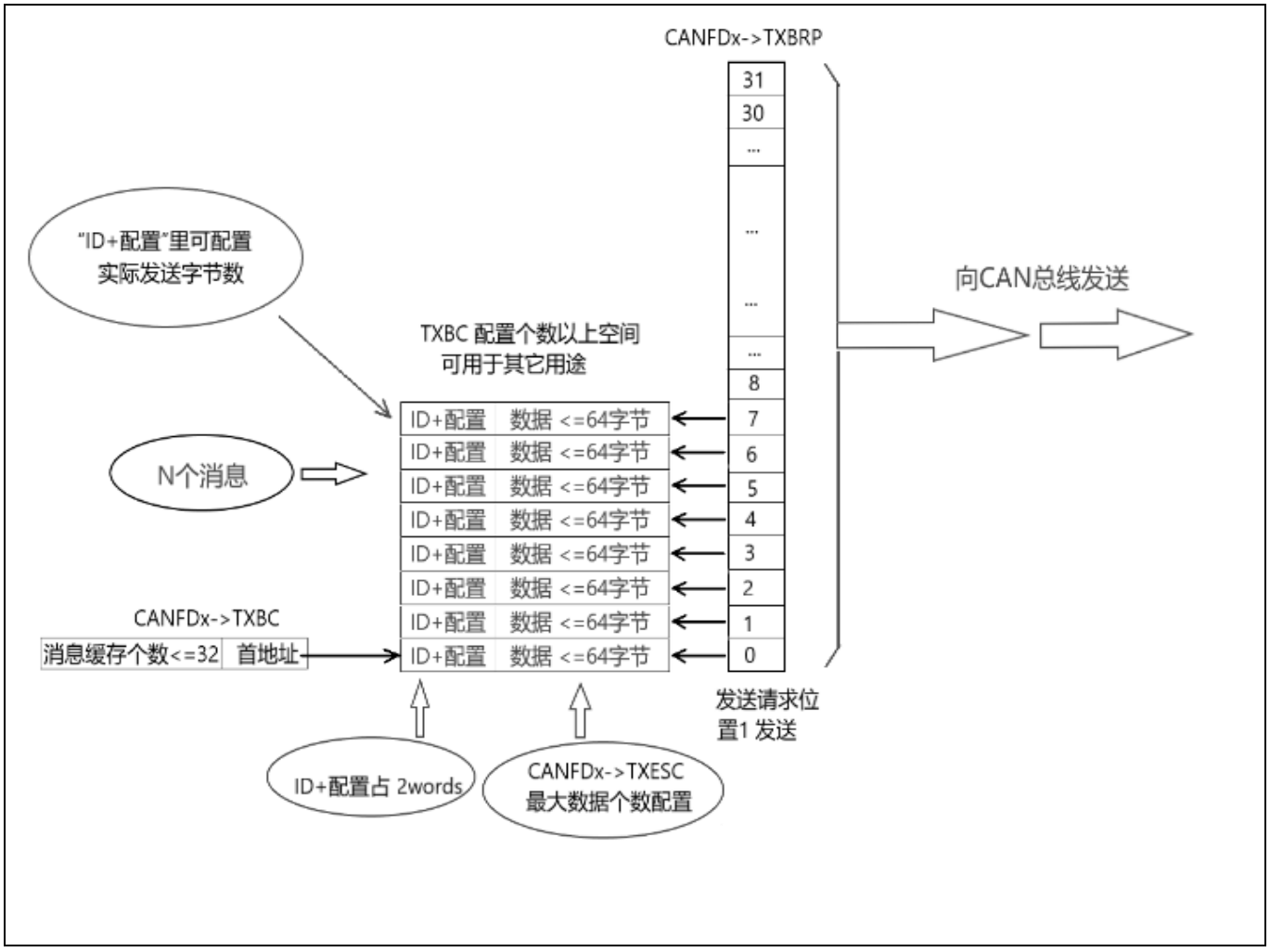

图 1-2 与发送报文相关的寄存器

本代码先配置 CANFD 的位率寄存器，再配置上述发送报文信息，最后，周期性的让 CANFDx->TXBRP 对应位置 1，启动报文的周期性发送。

1.2 执行结果

本代码在 NuMaker-M467HJ V1.0 板上测试，CANFD 接口连续发出报文，可用另一块 NuMaker-M467HJ 板子接收报文，并把收到的报文打印出来查看。本代码循环发送 8 个不同格式的报文，如图 1-3。

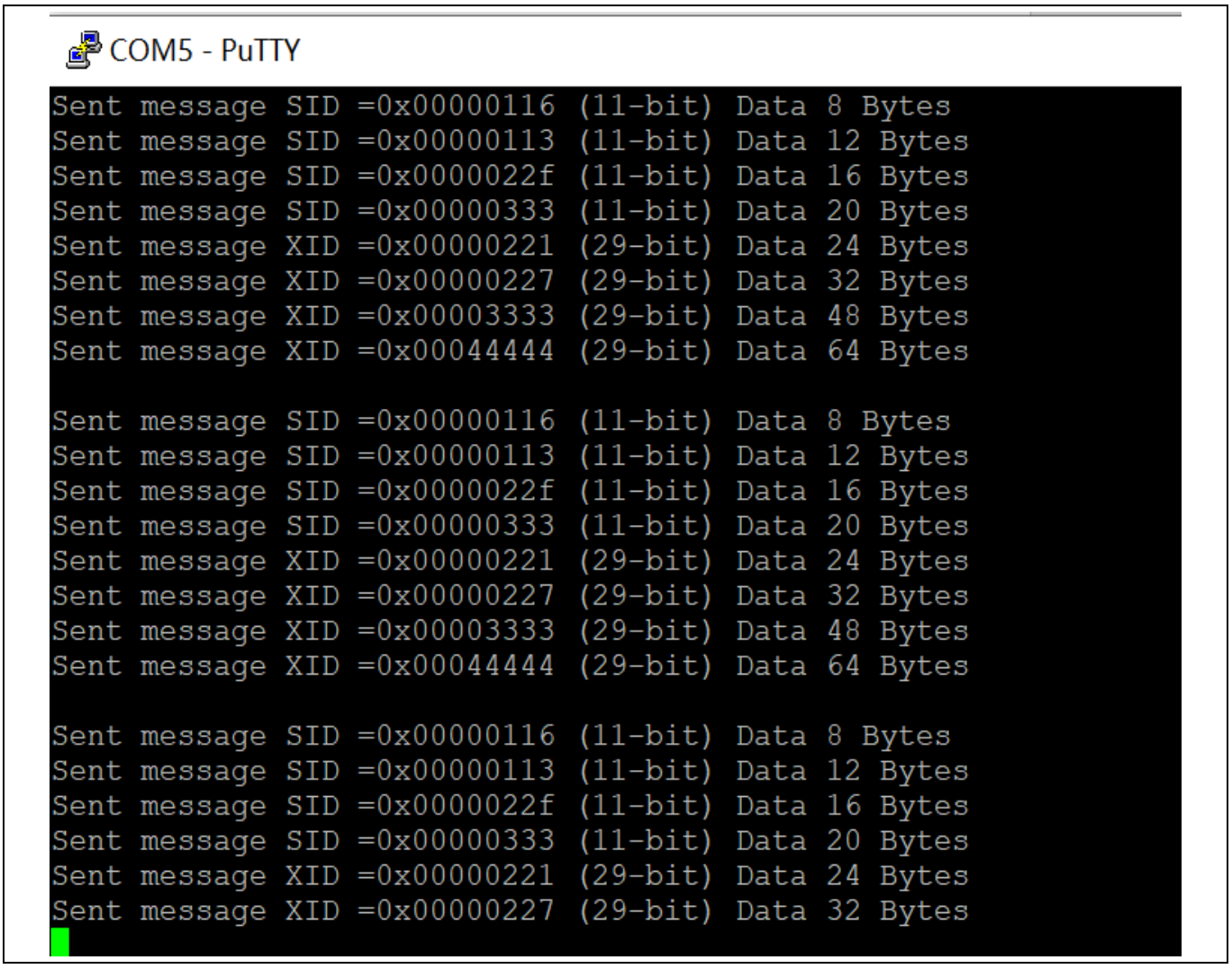

图 1-3 执行结果，打印信息

用另一个 NuMaker-M467HJ 板子的 CANFD 接收打印报文，这个板子执行代码：M460_Series_BSP_CMSIS_V3.00.001\SampleCode\StdDriver\CANFD_CANFD_TxRxINT，注意调整代码中的 CANFD 接口到 PJ10,PJ11 引脚，就可打印出如图 1-4 报文信息。

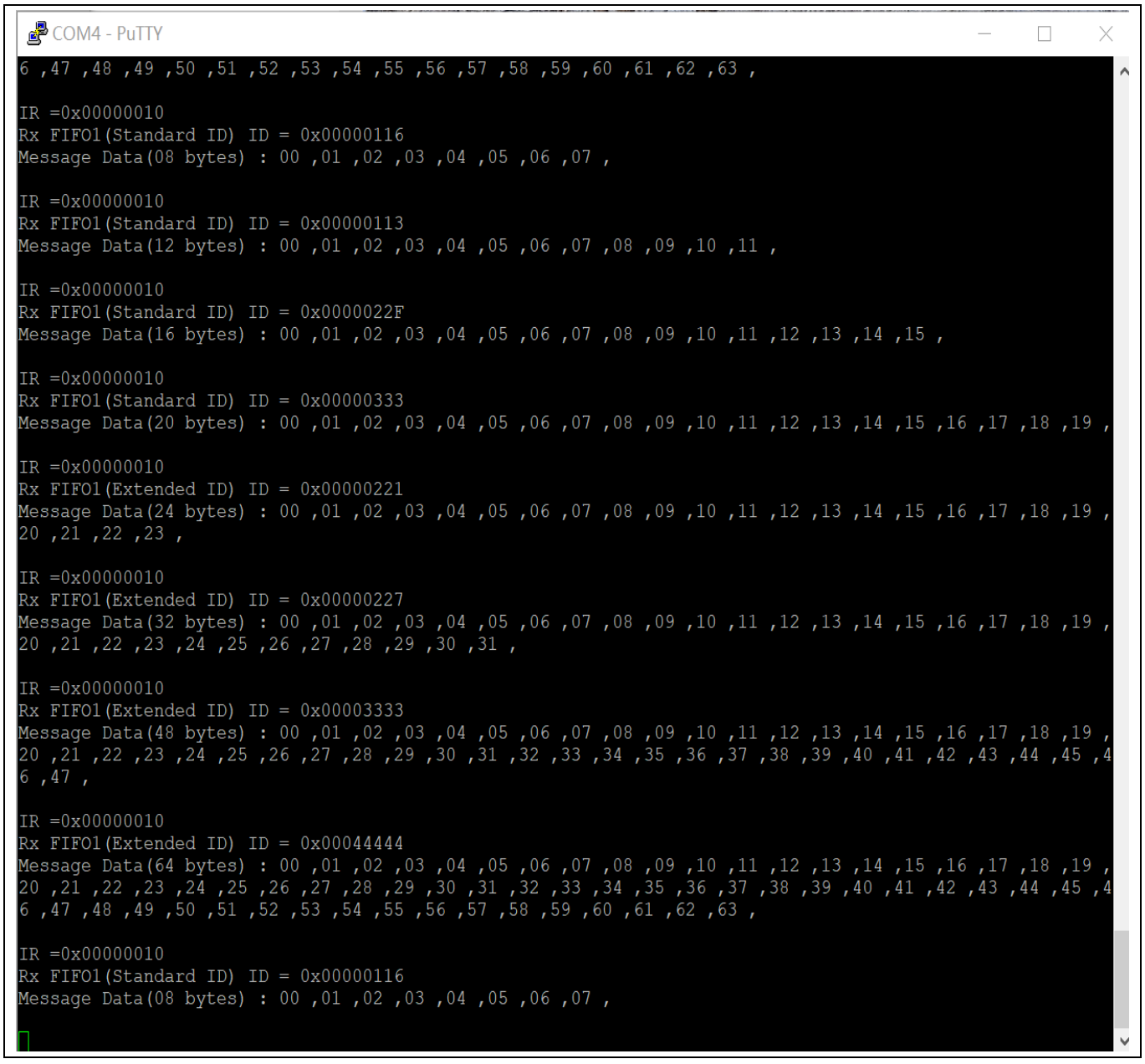

图 1-4 CANFD 总线上的消息

2. 代码介绍

配置 CANFD 传输速率，代码位于 *main.c*。

```
/// input: psCanfd, The value must be CANFD0,CANFD1,CANFD2 or CANFD3
void CANFD_BitRate_Init(CANFD_T *psCanfd)
{                                           // CANFD_CLk = 200M/5 = 40MHz
    psCanfd->NBTP = (3 << 25) +             // NSJW  = 3+1 =4  CLK
                    (1 << 16) +             // NBRP  = 1+1 =2 ,     // prescaler = 2
                    (13 << 8) +             // NTSG1 =13+1=14 CLK  // One bit =1+14+5 =20 CLK
                    (5 - 1);                // NTSG2 = 5  CLK

    psCanfd->DBTP = ((1 - 1) << 16) +       // DBRP  = 1 prescaler
                    ((6 - 1) << 8) +        // DTSG1 = 6 CLK       // One bit = 1+6+3 =10 CLK
                    ((3 - 1) << 4) +        // DTSG2 = 3 CLK
                    (2 - 1);                // DSJW  = 2CLK   采样点左右调整的最大时钟数
}
```

CANFD 接收时，数据域在 DTSG1 与 DTSG2 的中间采样信号电平，其它域时钟个数定义用 NTSG1 和 NTSG2。一位的时钟总数需再加一个时钟的同步段。如图 2-1。

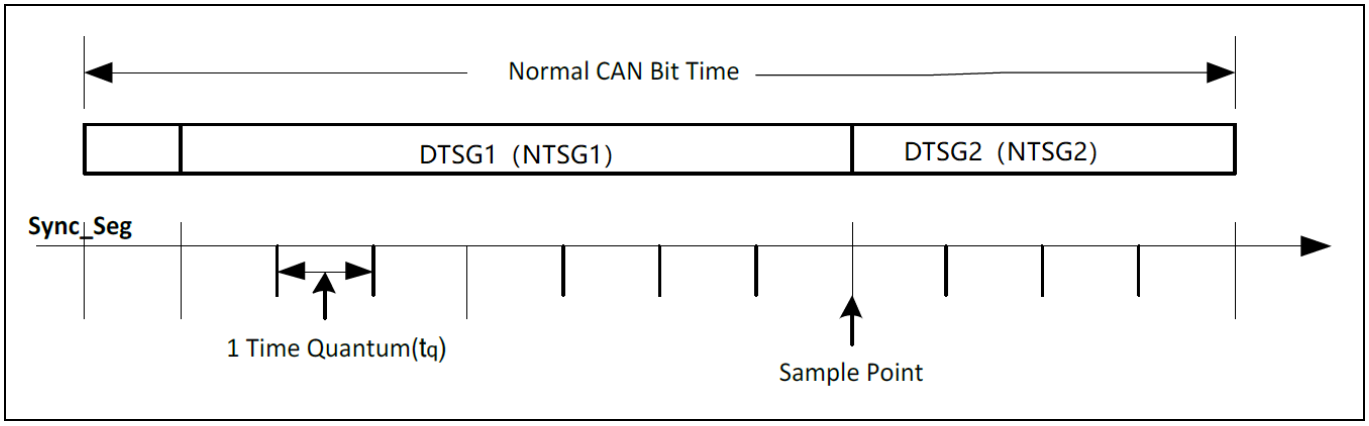

图 2-1 CANFD 一位时间的定义

发送报文开头的 2words 配置消息 ID、数据长度等，这部分由函数 CANFD_Tx_Init()先配置好，数据域在报文发送前实时更新(当然发送报文前，也可以更新这 2words 包头)。

```
    // There configured 8 Txbuff elements
void CANFD_Tx_Init(CANFD_T *psCanfd, int16_t TxBuff_Quantity, int16_t DLC_Max, uint16_t
TxBuff_StartAddr)
{
    psCanfd->TXBC = (TxBuff_Quantity << CANFD_TXBC_NDTB_Pos)
                    + TxBuff_StartAddr ;                    // Start address of messages RAM

    psCanfd->TXESC = DLC_Max <<  CANFD_TXESC_TBDS_Pos;       // 7 mean 64 Bytes

//  u32Config = 0x00300000 mean EFC=0，FDF=1, BRS=1 // If DLC <8,the Byte quantiry =DLC
```

```
    CANFD_TxBuff[0].u32Id       = 0x111 << 18 ;                   // SID = 0x111
    CANFD_TxBuff[0].u32Config   = 0x00300000 + 0x00080000 ;      // DLC=8 = 8 Bytes
//  CANFD_TxBuff[0].u32Config = 0x00300000 + 0x00050000 ;  // It has DLC data if(DLC <=8)

    CANFD_TxBuff[1].u32Id       = 0x113 << 18 ;                   // SID = 0x113
    CANFD_TxBuff[1].u32Config   = 0x00300000 + 0x00090000 ;      // DLC=9 = 12 Bytes
    CANFD_TxBuff[2].u32Id       = 0x22F << 18 ;                   // SID = 0x22F
    CANFD_TxBuff[2].u32Config   = 0x00300000 + 0x000A0000 ;      // DLC=A = 16 Bytes
    CANFD_TxBuff[3].u32Id       = 0x333 << 18 ;                   // SID = 0x333
    CANFD_TxBuff[3].u32Config   = 0x00300000 + 0x000B0000 ;      // DLC=B = 20 Bytes

    CANFD_TxBuff[4].u32Id       = 0x40000000 + 0x221 ;            // XID = 0x221
    CANFD_TxBuff[4].u32Config   = 0x00300000 + 0x000C0000 ;      // DLC=C = 24 Bytes
    CANFD_TxBuff[5].u32Id       = 0x40000000 + 0x227 ;            // XID = 0x227
    CANFD_TxBuff[5].u32Config   = 0x00300000 + 0x000D0000 ;      // DLC=D = 32 Bytes
    CANFD_TxBuff[6].u32Id       = 0x40000000 + 0x3333 ;           // XID = 0x3333
    CANFD_TxBuff[6].u32Config   = 0x00300000 + 0x000E0000 ;      // DLC=E = 48 Bytes
    CANFD_TxBuff[7].u32Id       = 0x40000000 + 0x44444 ;          // XID = 0x44444
    CANFD_TxBuff[7].u32Config   = 0x00300000 + 0x000F0000 ;      // DLC=F = 64 Bytes
}
```

最多可配置 32 条等待发送的报文，如果想启动某条报文的发送，只要在 CANFD 空闲时，把寄存器 TXBAR 的对应位置 1 就可以了。

```
 while (CANFD_GET_COMMUNICATION_STATE(psCanfd) != eCANFD_IDLE)
 {
   if (--u32TimeOutCnt == 0) break;            // waiting for CANFD idle
 }

 psCanfd->TXBAR = 1 << 3 ;       // Tx_Element_Buff 3 Request transmitting
```

3. 软件与硬件需求

3.1 软件需求

- BSP 版本
  - M460_Series_BSP_CMSIS_V3.00.001
- IDE 版本
  - Keil uVersion 5.38

3.2 硬件需求

- 电路组件
  - NuMaker-M467HJ V1.0
- 示意图
  - 将 两个 NuMaker-M467HJ 板的 CANFD 接口连接，以显示范例代码的执行结果。

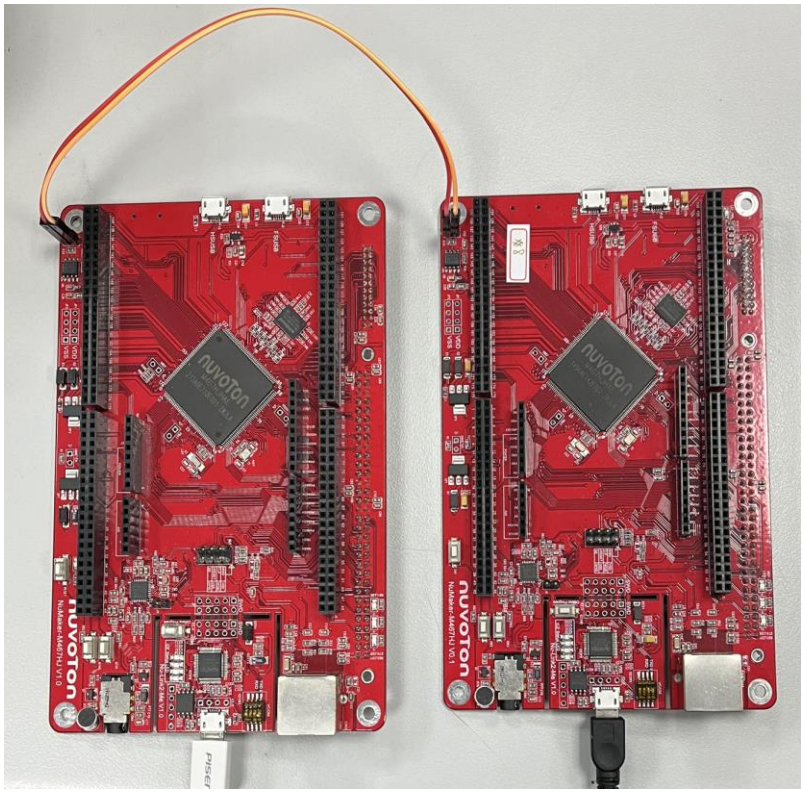

图 3-1 两个 NuMaker-M467HJ 板的 CANFD 接口连接

4. 目录信息

EC_M460_CANFD_Tx_Simply _V1.00	
Library	Sample code header and source files
CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
SampleCode	
ExampleCode	Source file of example code

图 4-1 目录信息

5. 范例程序执行

1. 根据目录信息章节进入 ExampleCode 路径中的 KEIL 文件夹，双击 *EC_M460_CANFD_Tx_Simply.uvprojx*。
2. 进入编译模式界面
   - 编译
   - 下载代码至内存
   - 进入 / 离开仿真模式
3. 进入仿真模式界面
   - 执行代码

6. 修订纪录

Date	Revision	Description
2023.02.20	1.00	初始发布。

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.