Firmware upgrade through CAN bus

Application Note for Cortex®-M0/M4 Series

Document Information

Abstract	本文件介绍如何使用CAN进行程序更新
Apply to	具有CAN接口的NuMicro® Cortex®-M0/M4全系列，本文以NuMicro® M451系列为例。

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1 概述

CAN Bus(Controller Area Network)是一种串行双线式全双工的通讯规格，由德国Bosch 于1985年提出，后来成为国际化的标准通讯接口(ISO11898)，普遍应用于车辆内部各种传感器或组件间的数据通讯，采用CAN-bus能大幅缩减通讯线缆的使用量，相对的也减少了许多线路上的接点，逐渐也被用于其它领域 如: 航空、航海电子仪器、工厂自动化、电梯、医疗仪器。

ISP(In System Program)是指「系统编程」，目标芯片使用USB/UART/SPI/I^2C/RS-485/CAN周边接口的LDROM，引导代码去更新芯片内部APROM、 DataFlash。

若产品需要修正程序错误或是新增功能，常需进行程序更新，来延长产品的生命周期，用户可以透过CAN Bus进行程序更新的动作。至此，本文将详述如何在新唐M451系列使用CAN ISP功能，实现程序更新。

1.1 CAN 介绍

C_CAN由CAN内核，报文RAM，报文处理器，控制寄存器和模块接口组成。CAN内核按照CAN协议2.0版本A部分和B部分规范执行通信。位速率最高可达 1Mbit/s。为与物理层相连，还需另外外接硬件收发器。

在CAN网络中，各个报文对象是可以独立配置的。报文对象和用于在接收时进行报文过滤的标识符掩码都存储在报文RAM中。所有与报文处理相关的功能都在报文处理器中执行。这些功能包括接收过滤、CAN内核与报文RAM之间的报文传输、处理传送请求以及模块中断的产生。

C-CAN的寄存器组可以通过模块接口被软件直接访问。这些寄存器用来控制/配置CAN内核和报文处理器，以及访问报文RAM。

1.1.1 CAN 框图

C_CAN与AMBA APB总线相连，如下图1-1所示

- **CAN 内核**
  包括 CAN 协议控制器和用于报文串/并数据转换的 RX/TX 移位寄存器
- **报文 RAM**
  用于保存报文对象和标识符掩码
- **寄存器组**
  包含所有控制和配置 C_CAN 的寄存器
- **报文处理器**
  用于控制 CAN 内核的 RX/TX 移位寄存器和报文 RAM 间的数据传输的状态机，以及按照控制和配置寄存器中设定产生中断
- **模块接口**
  C_CAN 与来自 CPU 的 AMBA APB16-位总线相接

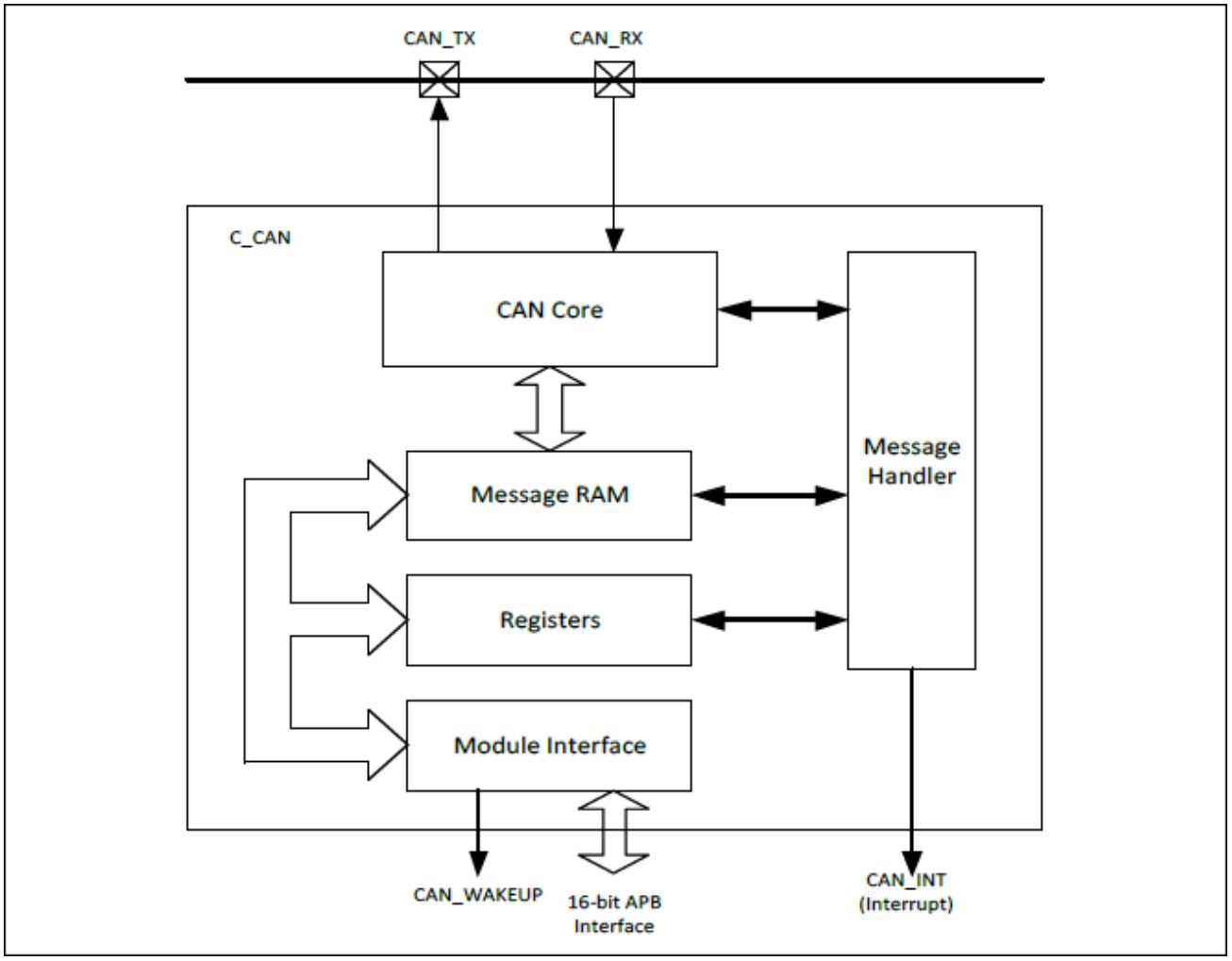

CAN_TX
CAN_RX
C_CAN
CAN Core
Message RAM
Registers
Module Interface
Message Handler
CAN_WAKEUP
16-bit APB Interface
CAN_INT (Interrupt)

图1-1 CAN 外设框图

1.1.2 CAN Transceivers

控制器区域网络（CAN）收发器兼容ISO 11898高速物理层标准的规范。专门为这些装置而设计，速率高达1兆比特每秒（Mbps），并提供设备和CAN可靠的数据传输。

MCU CAN控制器（TXD）的输出连接到收发器驱动器输入引脚D，CAN控制器（RXD）的输入连接到收发器驱动器输出引脚R，收发器通过CANH和CANL引脚连接到差分总线。

通常情况下，终端在总线的每一端是一个120Ω的电阻。总线是一对双绞线，标准半双工多点拓扑结构中，其阻抗特性为120Ω。总线的每一端都采用符合标准的120Ω电阻进行端接，以尽量减少信号反射在总在线。

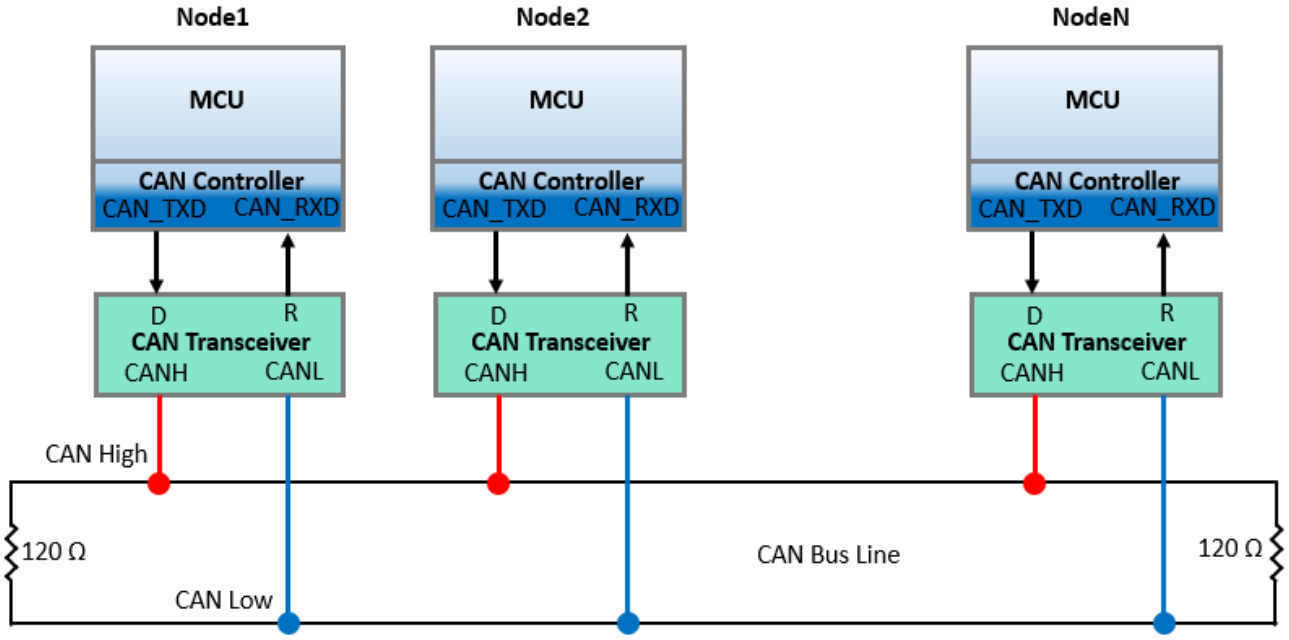


图1-2 CAN 总线系统

1.2 ISP 介绍

FMC 功能包括存储器组织，启动选择，IAP， ISP， 片上flash编程，和checksum 计算。片上存储器是可编程的，包括APROM，LDROM，数据Flash和用户配置区。地址映像包括flash存储映像和5个地址映像：支持IAP功能的LDROM，不支持IAP功能的LDROM，支持IAP功能的APROM， 不支持IAP功能的APROM，以及支持IAP功能的Boot Loader。

LDROM 是用于通过引导程序，来执行在系统编程(ISP)的功能。LDROM是4KB的片上flash存储器，flash地址是从0x0010_0000到0x0010_0FFF。APROM是用户应用程序的主要存储器。所有片上flash存储器, 页擦除的大小是2KB。

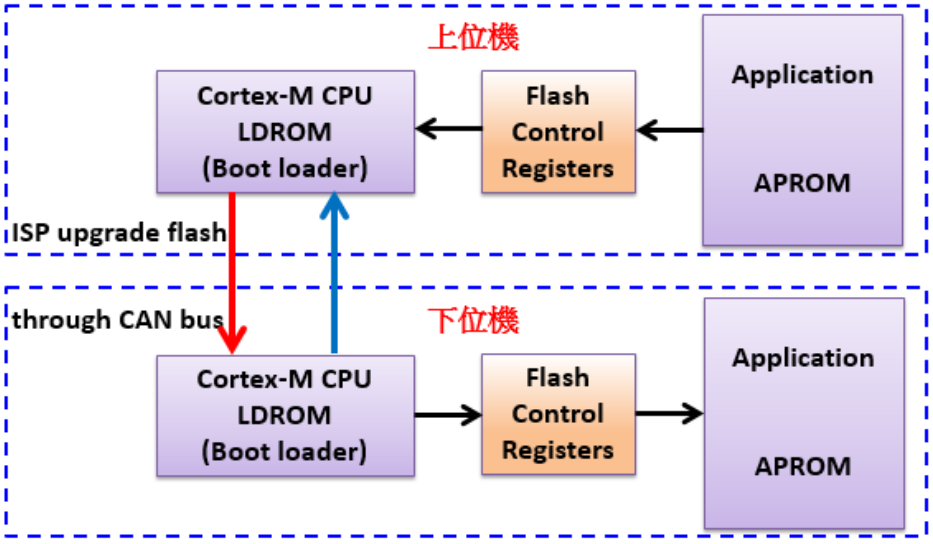


图1-3 CAN ISP 系统框图

1.2.1 ISP 功能

NuMicro® M451系列支持在系统编程(ISP)模式，可以通过软件控制重新烧写片上flash。使用ISP可以在目标板上直接编程，与多种接口相搭配，如UART，USB，I2C，SPI和 CAN (依据各芯片特性) 。使得用户可以使用多种方法来更新APROM的代码。

- ✧ 支持 flash 页擦除功能
- ✧ 支持数据 flash 编程
- ✧ 支持读数据 flash 功能
- ✧ 支持读公司 ID 功能
- ✧ 支持读设备 ID 功能
- ✧ 支持读 UID 功能
- ✧ 支持内存 checksum 计算功能
- ✧ 支持系统内存向量重映像功能

1.2.2 ISP 流程

FMC控制器提供了片上flash内存的读，擦除和编程操作。一些FMC控制器的寄存器是写保护的，所以在设定之前要解锁。

在解锁了保护寄存器之后，用户需要设定FMC_ISPCTL控制寄存器来决定更新LDROM，APROM或配置区，然后设定ISPEN (FMC_ISPCTL[0]) 来使能ISP功能。

一旦FMC_ISPCTL寄存器被设置成功，用户可以设定FMC_ISPCMD(参考上述ISP命令行表)来完成相应的操作。设置FMC_ISPADDR作为flash内存的目标地址。FMC_ISPDAT可以作为设定数据来编程或作为读寄存器命令FMC_ISPCMD的返回数据。

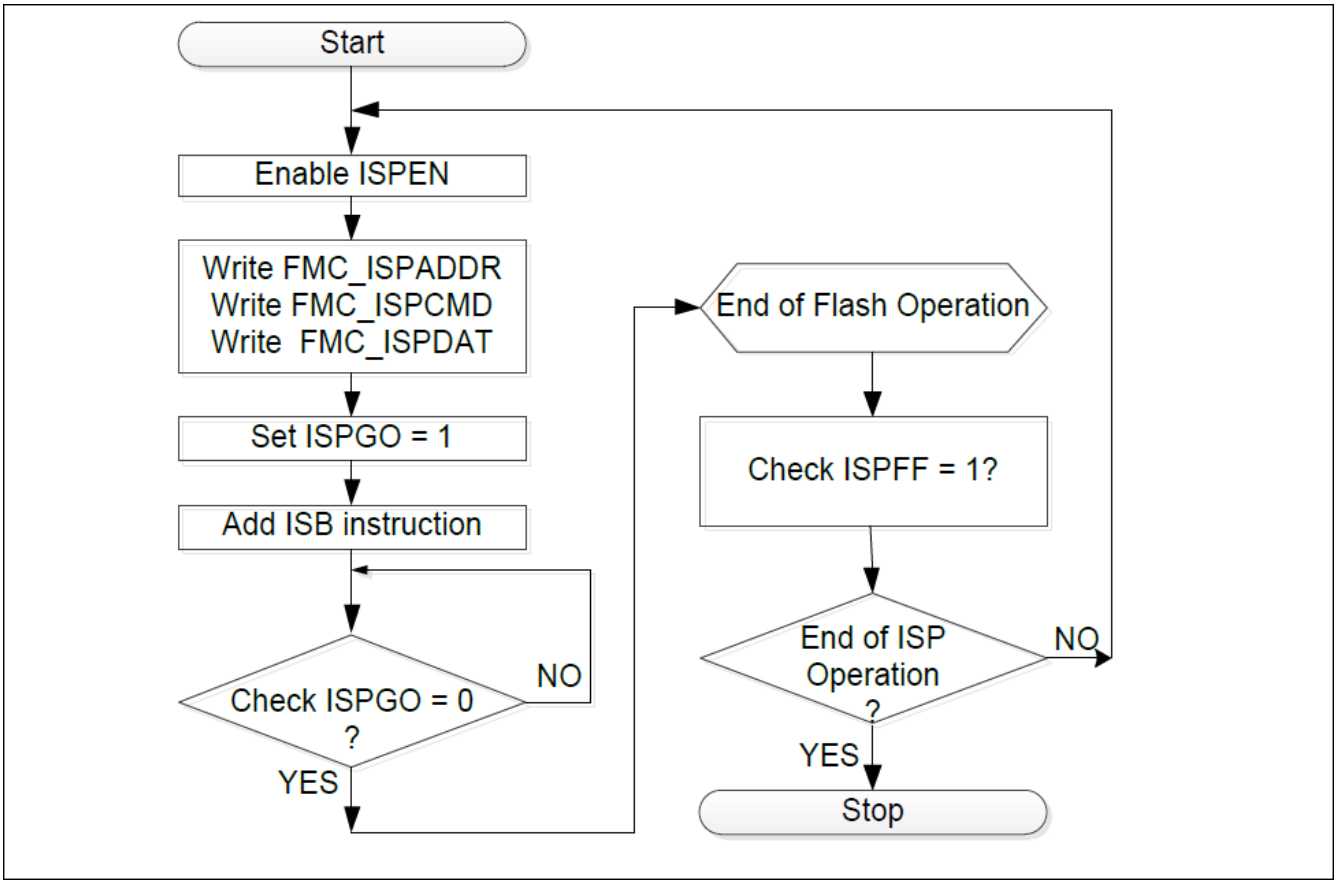

Start
Enable ISPEN
Write FMC_ISPADDR
Write FMC_ISPCMD
Write FMC_ISPDAT
Set ISPGO = 1
Add ISB instruction
Check ISPGO = 0 ?
NO
YES
End of Flash Operation
Check ISPFF = 1?
End of ISP Operation ?
NO
YES
Stop

图1-4 ISP 流程图

2 CAN ISP 范例应用

科技发达的世代许多领域，如航空、航海、电梯以及普及应用在车内部各种传感器都透过CAN达到组件间的数据通讯。

产品可以透过ISP，在线烧錄的功能，不需将 IC从机器中取下，放在特定的烧錄器进行烧錄的动作，可以直接透过CAN的传输接口，搭配特定的传输协议即可进行芯片抹除、写入，是一个便利的功能，来延长产品的生命周期。

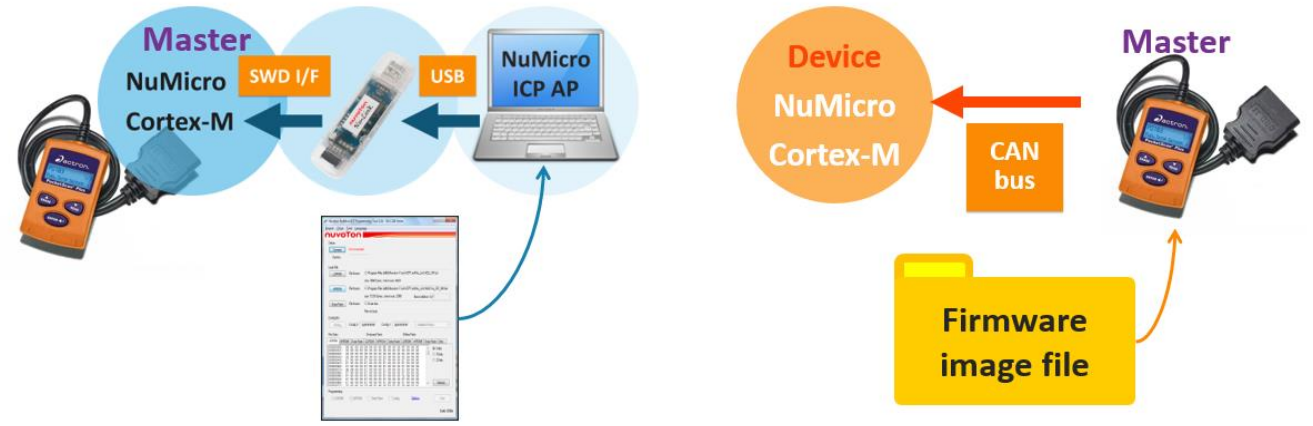


图2-1 CAN ISP 应用示意图

2.1 CAN ISP 架构

上位机上电执行LDROM之ISP功能，透过FMC读取APROM经由CAN bus将数据给下位机；下位机上电执行LDROM之ISP功能，经由CAN bus将数据接收并透过FMC写入APROM，达到更新程序。

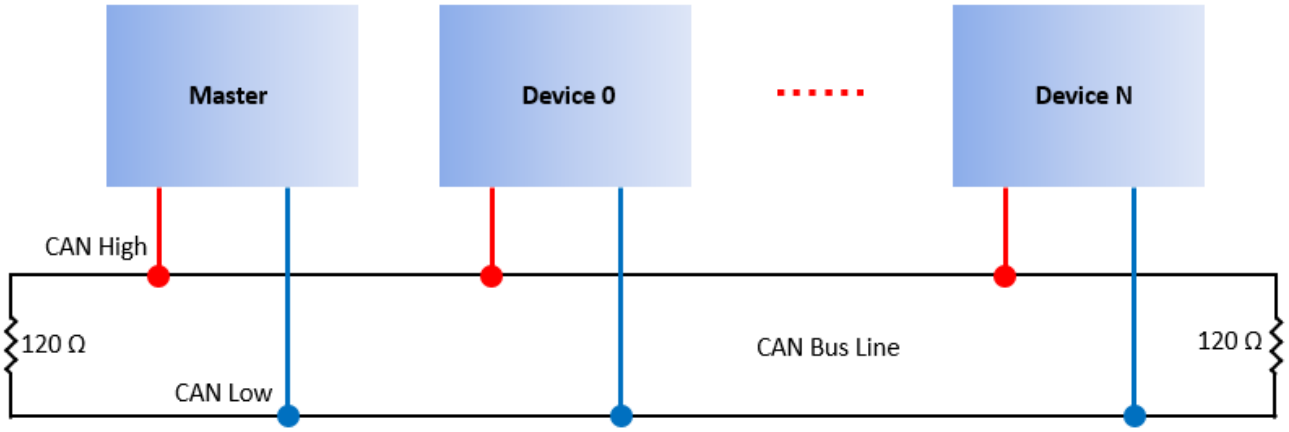


图2-2 CAN ISP更新韧体示意图

- 开发硬件为 : NuMaker PFM M453
- 上位机 LDROM 功能 : 读取上位机 APROM 透过 CAN bus 传输。
- 上位机 APROM 功能 : 三色 LED，亮紫光。
- 下位机 LDROM 功能 : 透过 CAN bus 接收并烧入 APROM。

2.2 CAN bus 通讯

CAN协议2.0版本分为A版和B版两种的通信规范，A版协议仅支持11位标识符（称为标准帧）通常称为CAN 2.0A，B版协议兼容11位，向上扩展到29位标识符（称为扩展帧）通常称为CAN 2.0B。本文应用于CAN ISP 更新程序，数据封包需较高的优先权使用CAN 2.0A通讯协议，如下图所示为此应用8字节数据协议，4个字节的地址，4个字节的数据。

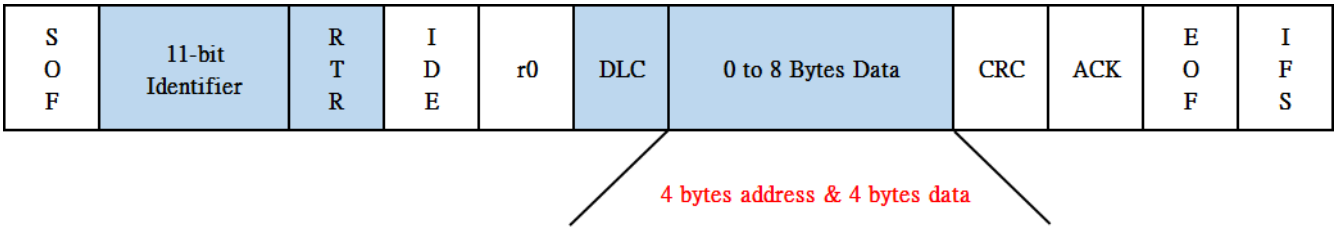


图2-3 CAN 2.0A 11 bits 协议

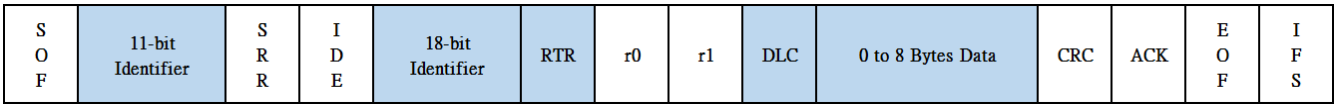


图2-4 CAN 2.0B 29 bits 协议

由于CAN bus在传送数据的过程是类似广播的方式，所以在node数量众多的时候，单一node会收到所有node所传出的各种数据。若Master 传送标识符设为0x7FF；Device的接收标识符都设为0x7FF时，传送数据的方法就如同广播的方式，如下图所示。

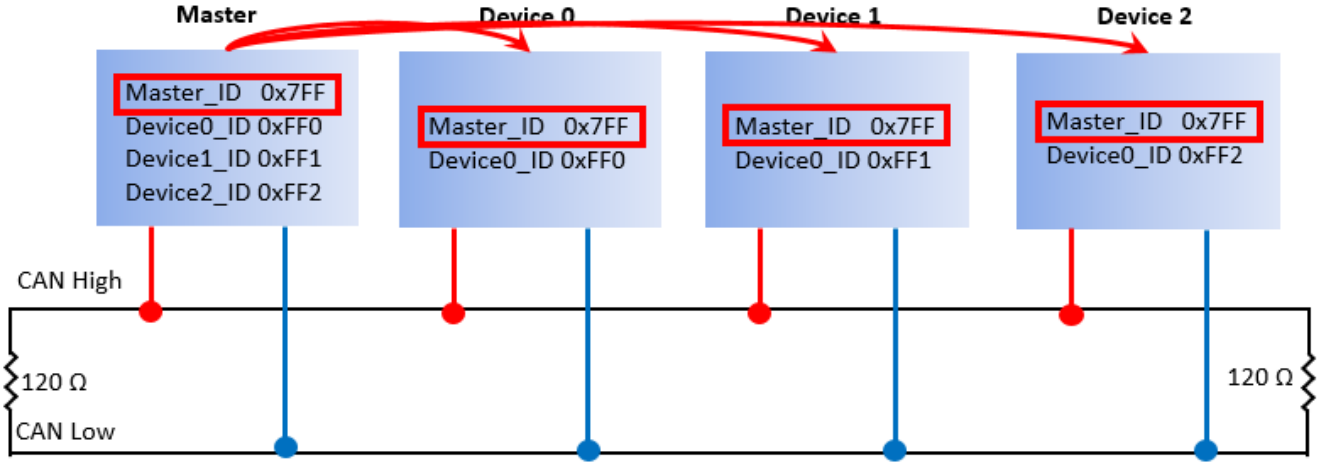


图2-5 CAN通讯

当Device 0要传送数据给Master时，Device传送标识符设为0xFF0；Master接收标识符都设为0xFF0，如下图所示。

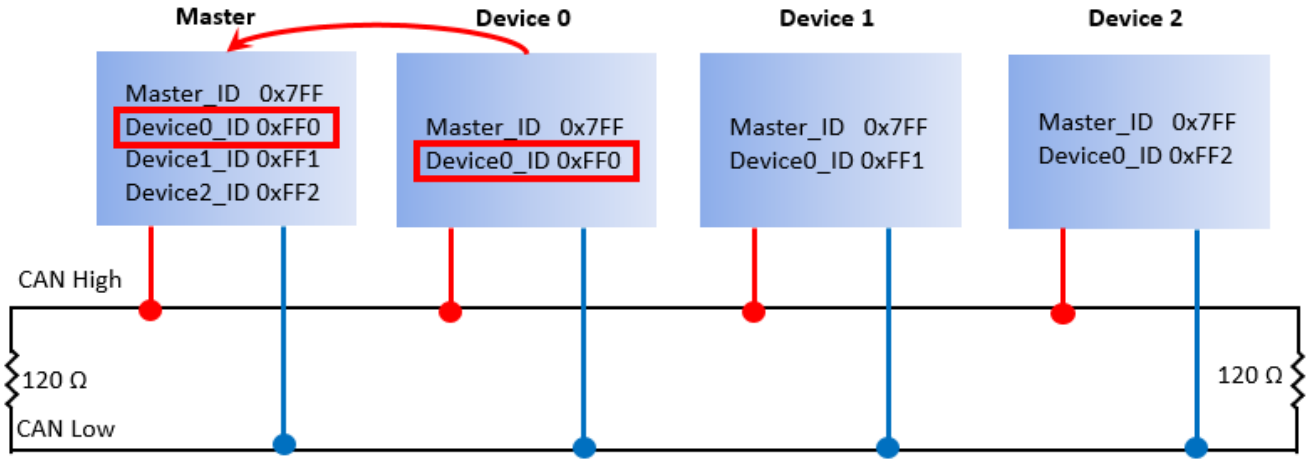


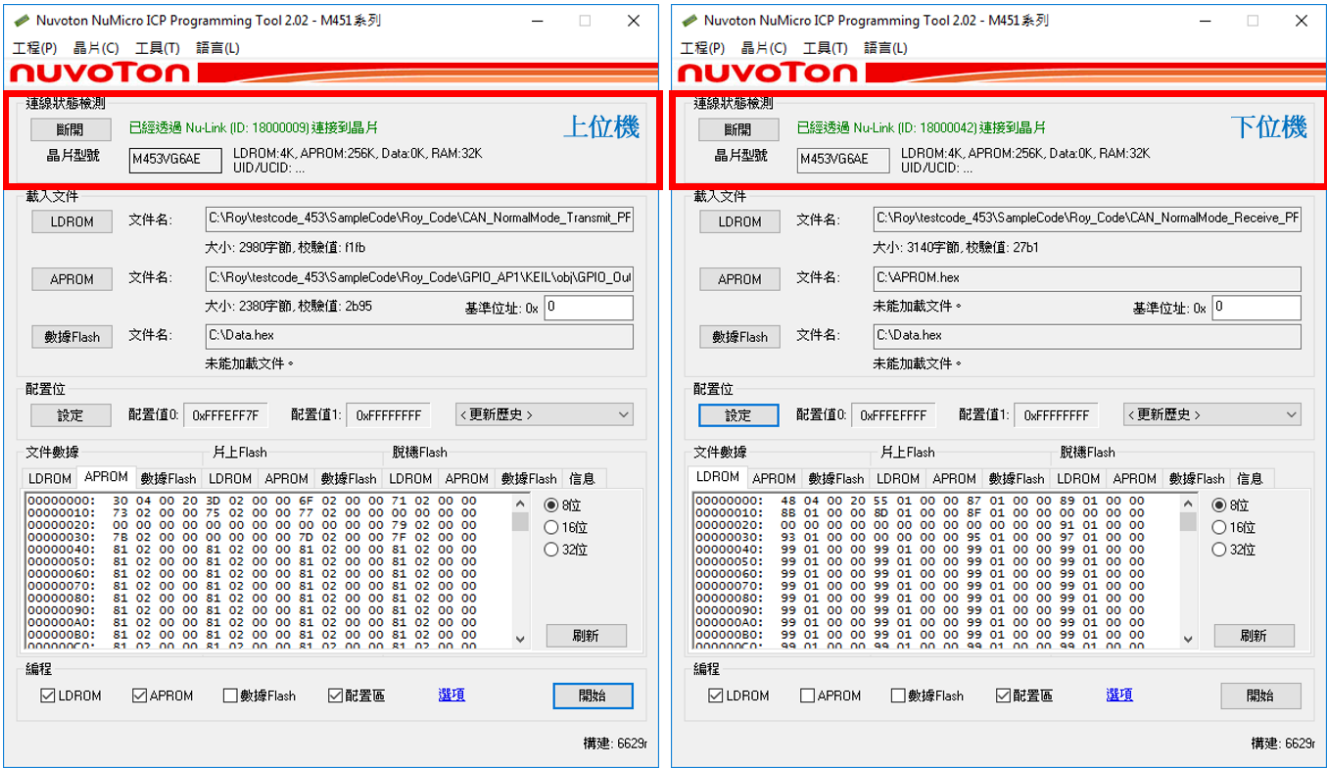
图2-6 CAN通讯

2.3 使用 ICP Tool

在量产的过程中需透过ICP tool将韧体烧入IC，同时也将ISP韧体程序烧入IC 的LDROM区块，这样往后产品若需要更新就可经由ISP功能进行更新程序，此节介绍如何使用ICP tool 将ISP、APP烧入上位机、下位机。

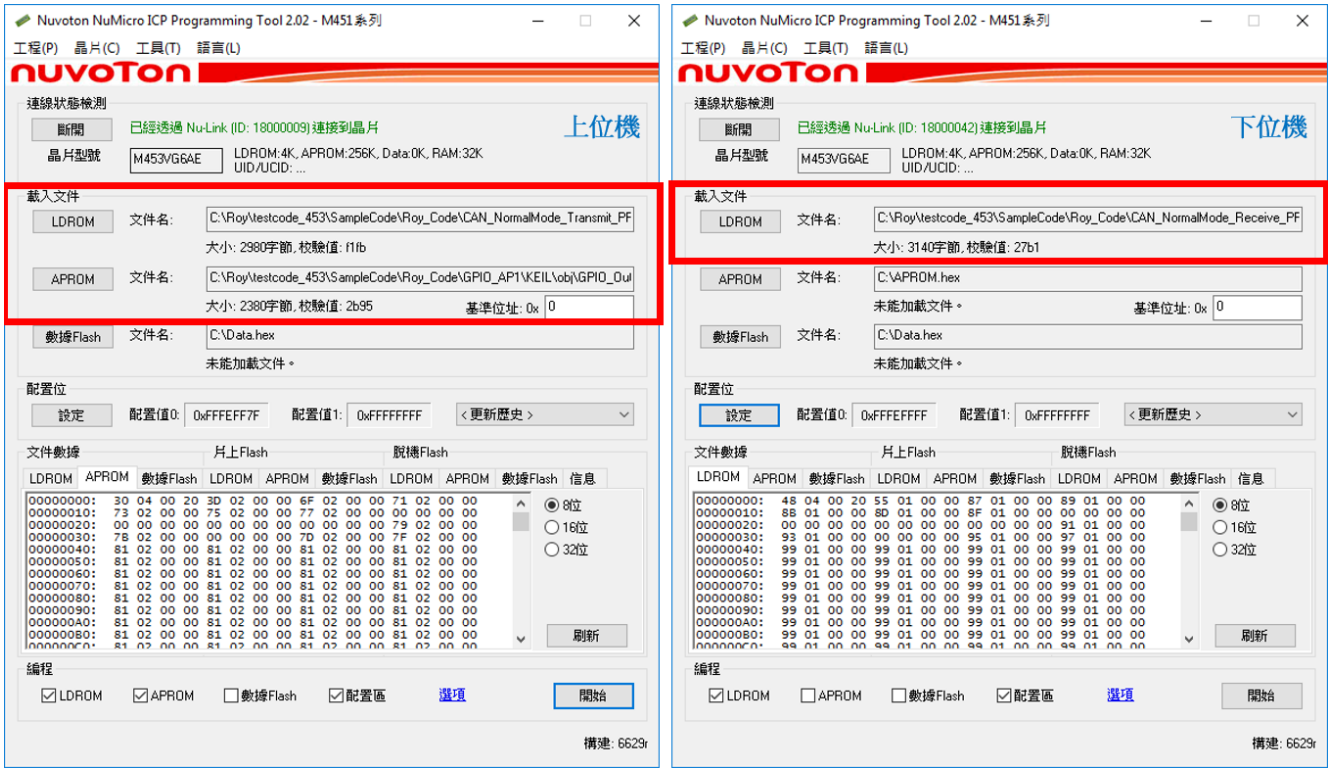
上位机需要透过ICP tool 将上位机的ISP程序烧入LDROM，以及要更新给下位机的APP烧入APROM；下位机也需要透过ICP tool 将下位机的ISP程序烧入LDROM。

a) 开启 ICP Tool，并连接上位机、下位机。

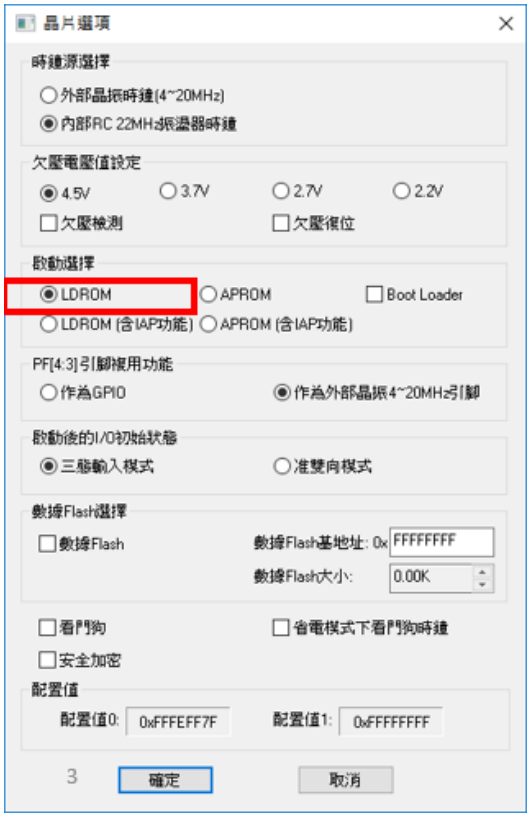


b) 载入 bin 檔

- 上位机 : 点击[LDROM] 加载上位机 ISP 程序放置 LDROM 「CAN_ISP_Master.bin」； 点击[APROM]加载要更新给下位机的 APP 程序放置 APROM。
- 下位机 : 点击[LDROM]加载下位机 ISP 程序放置 LDROM 「CAN_ISP_Device.bin」。

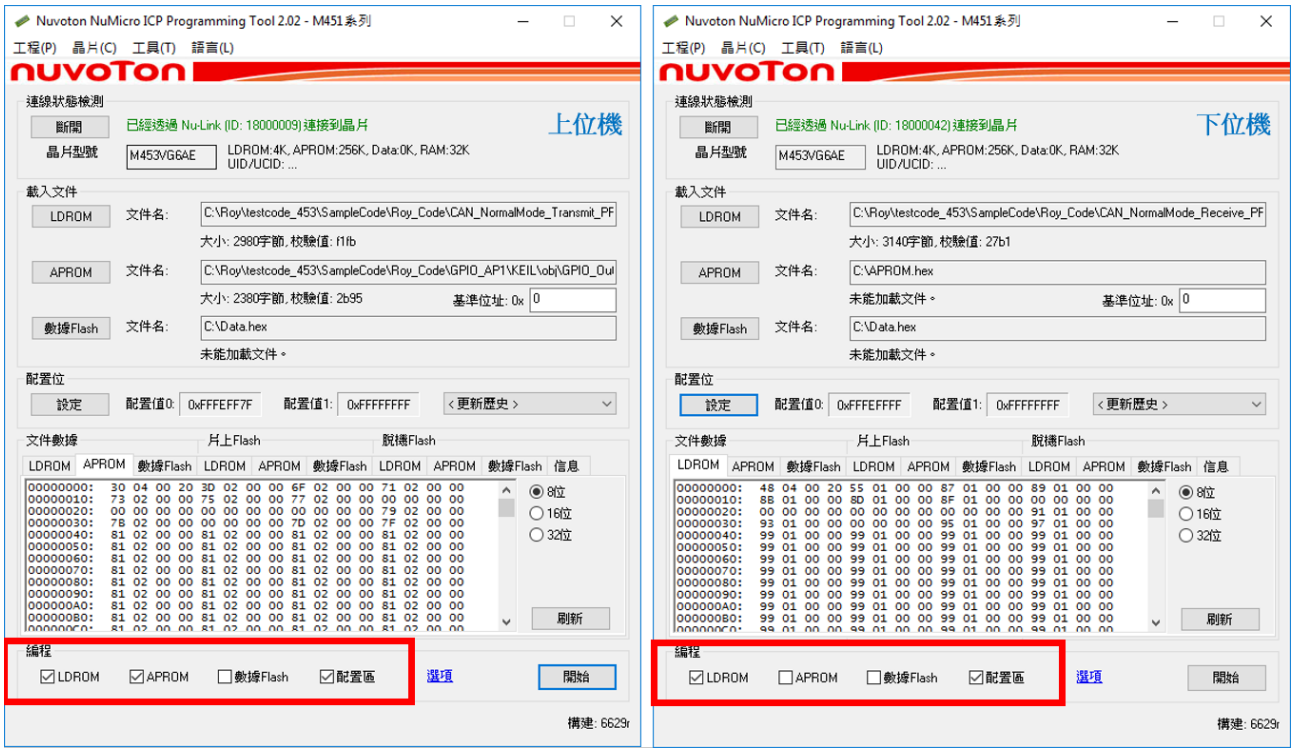


c) 上位机和下位机的 ICP tool 画面点击[设定]，进入设定画面，将[启动选择]设定为[LDROM]，点击确定。



d) 编程设定

- 上位机：将需要编程的区域选择为[LDROM、APROM、配置区]。
- 下位机：将需要编程的区域选择为[LDROM、配置区]。



e) ICP tool [选项]，勾选操作中的[擦除]、[烧写]、[验证]、[烧写后重启芯片]，点击确定。

f) 点选[开始]进行刻录。


Nuvoton NuMicro ICP Programming Tool 2.02 - M451系列
工程(P) 晶片(C) 工具(T) 語言(L)
連線狀態檢測
斷開
已經透過 Nu-Link (ID: 18000009)連接到晶片
上位機
晶片型號
M453VG6AE
LDROM:4K, APROM:256K, Data:0K, RAM:32K
UID/UCID: ...
載入文件
LDROM
文件名:
C:\Roy\testcode_453\SampleCode\Roy_Code\CAN_NormalMode_Transmit_PF
大小: 2980字節, 校驗值: f1fb
APROM
文件名:
C:\Roy\testcode_453\SampleCode\Roy_Code\GPIO_AP1\KEIL\obj\GPIO_Out
大小: 2380字節, 校驗值: 2b95
基準位址: 0x 0
數據Flash
文件名:
C:\Data.hex
未能加載文件。
配置位
設定
配置值0: 0xFFFEFF7F
配置值1: 0xFFFFFFFF
< 更新歷史 >
文件數據
片上Flash
脫機Flash
LDROM APROM 數據Flash LDROM APROM 數據Flash LDROM APROM 數據Flash 信息
00000000: 30 04 00 20 3D 02 00 00 6F 02 00 00 71 02 00 00
00000010: 73 02 00 00 75 02 00 00 77 02 00 00 00 00 00 00
00000020: 00 00 00 00 00 00 00 00 00 00 00 00 79 02 00 00
00000030: 7B 02 00 00 00 00 00 00 7D 02 00 00 7F 02 00 00
00000040: 81 02 00 00 81 02 00 00 81 02 00 00 81 02 00 00
00000050: 81 02 00 00 81 02 00 00 81 02 00 00 81 02 00 00
00000060: 81 02 00 00 81 02 00 00 81 02 00 00 81 02 00 00
00000070: 81 02 00 00 81 02 00 00 81 02 00 00 81 02 00 00
00000080: 81 02 00 00 81 02 00 00 81 02 00 00 81 02 00 00
00000090: 81 02 00 00 81 02 00 00 81 02 00 00 81 02 00 00
000000A0: 81 02 00 00 81 02 00 00 81 02 00 00 81 02 00 00
000000B0: 81 02 00 00 81 02 00 00 81 02 00 00 81 02 00 00
8位
16位
32位
刷新
編程
LDROM
APROM
數據Flash
配置區
選項
開始
構建: 6629r
Nuvoton NuMicro ICP Programming Tool 2.02 - M451系列
工程(P) 晶片(C) 工具(T) 語言(L)
連線狀態檢測
斷開
已經透過 Nu-Link (ID: 18000042)連接到晶片
下位機
晶片型號
M453VG6AE
LDROM:4K, APROM:256K, Data:0K, RAM:32K
UID/UCID: ...
載入文件
LDROM
文件名:
C:\Roy\testcode_453\SampleCode\Roy_Code\CAN_NormalMode_Receive_PF
大小: 3140字節, 校驗值: 27b1
APROM
文件名:
C:\APROM.hex
未能加載文件。
基準位址: 0x 0
數據Flash
文件名:
C:\Data.hex
未能加載文件。
配置位
設定
配置值0: 0xFFFEFFFF
配置值1: 0xFFFFFFFF
< 更新歷史 >
文件數據
片上Flash
脫機Flash
LDROM APROM 數據Flash LDROM APROM 數據Flash LDROM APROM 數據Flash 信息
00000000: 48 04 00 20 55 01 00 00 87 01 00 00 89 01 00 00
00000010: 8B 01 00 00 8D 01 00 00 8F 01 00 00 00 00 00 00
00000020: 00 00 00 00 00 00 00 00 00 00 00 00 91 01 00 00
00000030: 93 01 00 00 00 00 00 00 95 01 00 00 97 01 00 00
00000040: 99 01 00 00 99 01 00 00 99 01 00 00 99 01 00 00
00000050: 99 01 00 00 99 01 00 00 99 01 00 00 99 01 00 00
00000060: 99 01 00 00 99 01 00 00 99 01 00 00 99 01 00 00
00000070: 99 01 00 00 99 01 00 00 99 01 00 00 99 01 00 00
00000080: 99 01 00 00 99 01 00 00 99 01 00 00 99 01 00 00
00000090: 99 01 00 00 99 01 00 00 99 01 00 00 99 01 00 00
000000A0: 99 01 00 00 99 01 00 00 99 01 00 00 99 01 00 00
000000B0: 99 01 00 00 99 01 00 00 99 01 00 00 99 01 00 00
8位
16位
32位
刷新
編程
LDROM
APROM
數據Flash
配置區
選項
開始
構建: 6629r

3 范例程序

在2.2章节CAN bus 通讯中介绍CAN bus收发数据的方式包含一对一以及一对多的沟通方式，此应用文件提供一对一范例程序利于用户进入、开发；若要开发一对多的沟通方式，在3.2章节范例程序中会加以说明哪些部分是需要再增加的程序。

3.1.1 上位机

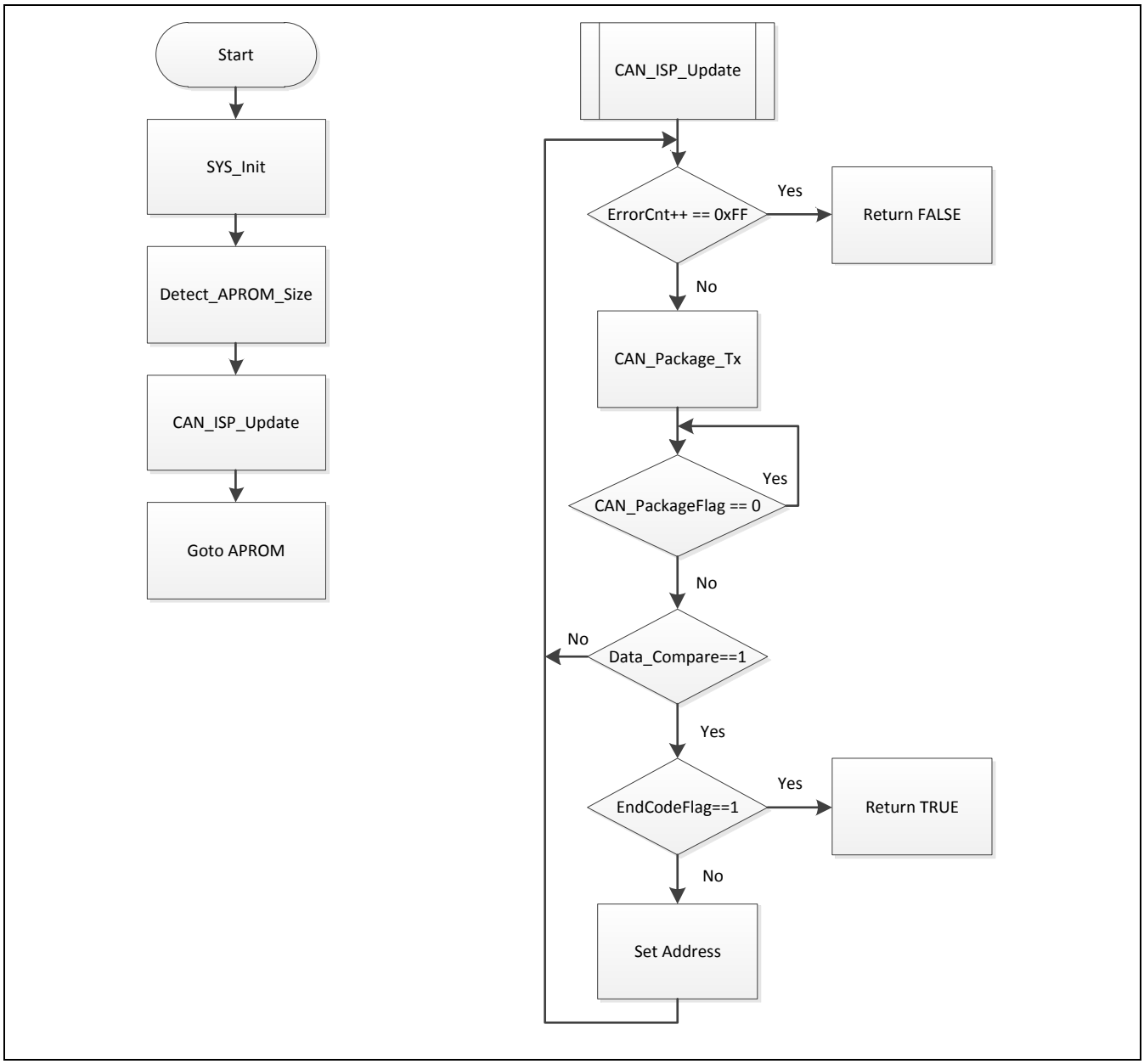


图3-1 上位机流程图

3.1.2 下位机

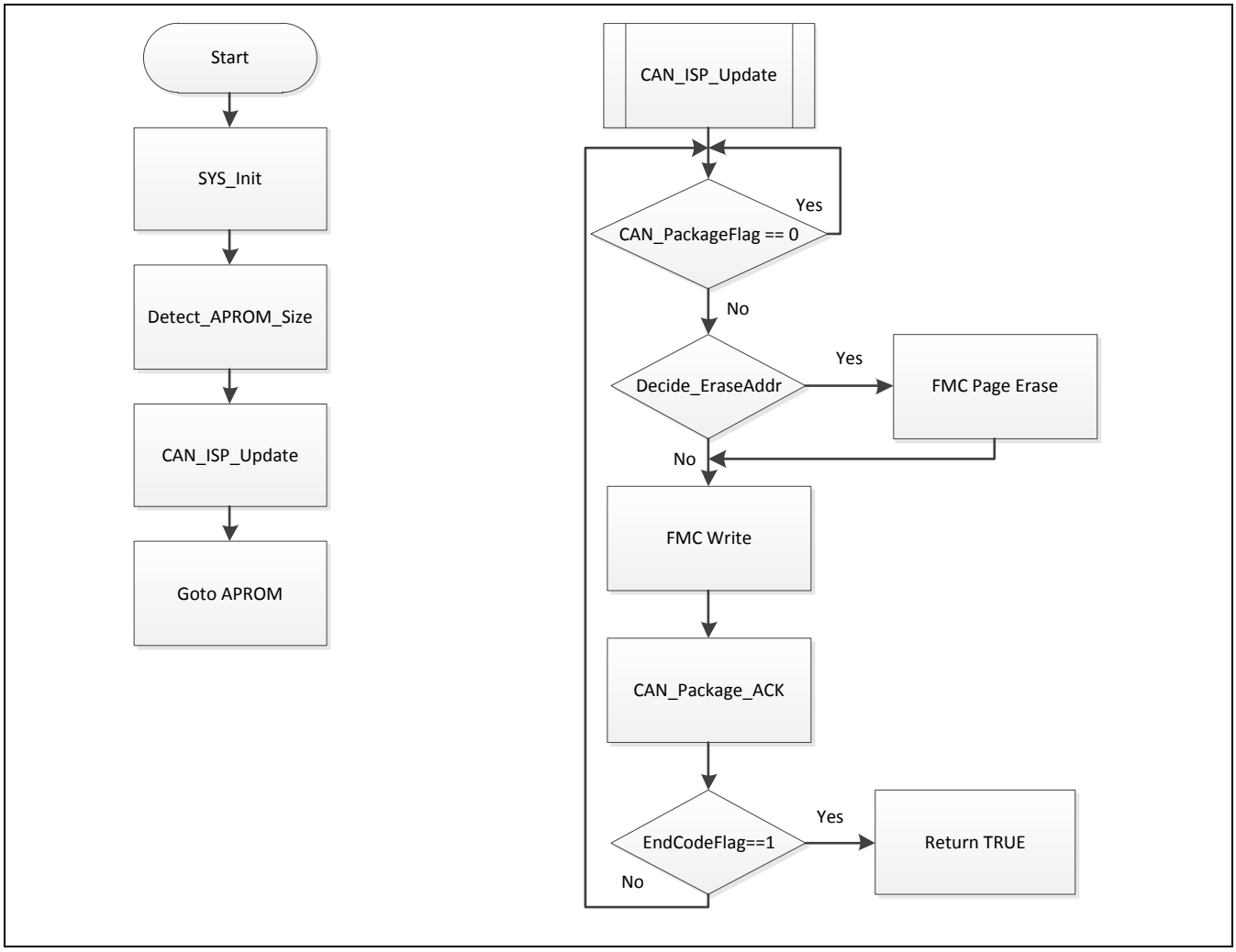


图3-2 下位机流程图

3.2 范例程序

3.2.1 上位机

3.2.1.1 上位机主程序

若需要一对多的沟通方式，上位机须要定义新的device的ID，如：#define Device1_ISP_ID 0x803。

```
#define PLL_CLOCK                       48000000
#define CAN_BAUD_RATE                   250000
#define Master_ISP_ID                   0x800
#define Device0_ISP_ID                  0x802
#define CAN_ISP_DtatLength              0x08
#define CAN_FailTime                    0xff
#define StartProgramAddress             0
int main(void)
{
    /* Unlock protected registers */
    SYS_UnlockReg();
    /* Init System, IP clock and multi-function I/O */
    SYS_Init();
    /* Init CAN port */
    can_init();
    /* Enable FMC ISP function */
    FMC_Open();
    if(APROM_detect()>0)
    {
        /* stat update program */
        if(CAN_ISP_Update(CAN0)>0)
        {
            goto _APROM;
        }
    }
    while(1);
    _APROM:
    /*clear bit*/
    outpw(&SYS->RSTSTS, 3);
    outpw(&FMC->ISPCTL, inpw(&FMC->ISPCTL) & 0xFFFFFFFC);
    outpw(&SCB->AIRCR, (V6M_AIRCR_VECTKEY_DATA | V6M_AIRCR_SYSRESETREQ));
    /* Trap the CPU */
    while(1);
}
```

3.2.1.2 上位机ISP更新程序

```
uint8_t CAN_ISP_Update(CAN_T *tCAN)
{
    STR_CANMSG_ISP CAN_buffer;

    uint8_t ErrorCnt=0, CompareFail_flag=1;
    memset(&CAN_buffer, 0, sizeof(CAN_buffer));
    CAN_buffer.Address = StartProgramAddress;

    while(1)
    {
        if(ErrorCnt++ == CAN_FailTime)              //ErrorCnt = 0xff CAN bus fail
        {
            return FALSE;
        }
        CAN_Package_Tx(CAN0, &CAN_buffer);          //send CAN ISP Package

        u8CAN_PackageFlag=0;
        TimeOut_flag=0;
        CLK_EnableSysTickINT(2000);                 // System Tick timer value us
        //Wait CAN ISP Package ACK
        while(u8CAN_PackageFlag == 0 && TimeOut_flag == 0);
        CLK_DisableSysTick();
        if(TimeOut_flag==0)
        {
            CompareFail_flag=1;
            //check CAN ISP Package
            if(((long)CAN_buffer.Address) != (*(long*)rrMsg.Data)) CompareFail_flag=0;
            if(CompareFail_flag)
            {
                //check CAN ISP EndCode
                if(CAN_buffer.Address == Chip_EndAddress)  return TRUE;
                ErrorCnt=0;
                CAN_buffer.Address += 4;            //for next address
            }
            else
            {
                if(CAN_buffer.Address < FMC_FLASH_PAGE_SIZE) CAN_buffer.Address =
StartProgramAddress;
                else CAN_buffer.Address = (CAN_buffer.Address / FMC_FLASH_PAGE_SIZE) *
FMC_FLASH_PAGE_SIZE;
```

```
            }
        }
    }
}
```

3.2.1.3 上位机传送、接收Message

若需要一对多的沟通方式，上位机须要新增Rx message的设定，哪一个device的数据要接收到报文的哪一个信道，报文信道可设定范围为0至31，共32个信道，如程序中批注一所示。

```
void CAN_Package_Tx(CAN_T *tCAN, STR_CANMSG_ISP *CANBuffer)
{
    STR_CANMSG_T tMsg;
    uint32_t APROM_Data=0;
    /* Send a 11-bit Standard Identifier message */
    tMsg.FrameType = CAN_DATA_FRAME;
    tMsg.IdType    = CAN_STD_ID;
    /* Setting Master ID */
    tMsg.Id        = Master_ISP_ID;
    tMsg.DLC       = CAN_ISP_DtatLength;
    /* Address*/
    memcpy(&tMsg.Data, &CANBuffer->Address, 4);
    APROM_Data = FMC_Read(CANBuffer->Address);
    /* Data*/
    memcpy(&tMsg.Data[4], &APROM_Data, 4);
    /* Configure Msg RAM and send the Msg in the RAM*/
    if(CAN_Transmit(tCAN, MSG(5), &tMsg) == FALSE)
    {
        return;
    }
}

uint8_t can_setRxMsg(CAN_T *tCAN)
{
    /* Receiving the data of the Device0_ISP_ID to channel0 of MSG SRAM */ 批注一
    if(CAN_SetRxMsg(tCAN, MSG(0), CAN_STD_ID, Device0_ISP_ID) == FALSE)
    {
        return FALSE;
    }
    return TRUE;
}
```

3.2.1.4 上位机中断程序

```
void CAN0_IRQHandler(void)
{
    uint32_t u8IIDRstatus;
    u8IIDRstatus = CAN0->IIDR;
    /* Check Status Interrupt Flag (Error status Int and Status change Int) */
    if(u8IIDRstatus == 0x00008000)
    {
        /*************************/
        /* Status Change interrupt*/
        /*************************/
        if(CAN0->STATUS & CAN_STATUS_RXOK_Msk)
        {
            /* Clear RxOK status*/
            CAN0->STATUS &= ~CAN_STATUS_RXOK_Msk;
        }

        if(CAN0->STATUS & CAN_STATUS_TXOK_Msk)
        {
            /* Clear TxOK status*/
            CAN0->STATUS &= ~CAN_STATUS_TXOK_Msk;
        }
    }
    else if(u8IIDRstatus!=0)
    {
        CAN_MsgInterrupt(CAN0, u8IIDRstatus);
        /* Clear Interrupt Pending */
        CAN_CLR_INT_PENDING_BIT(CAN0, (u8IIDRstatus - 1));
    }
}

void CAN_MsgInterrupt(CAN_T *tCAN, uint32_t u32IIDR)
{
    if(u32IIDR == 1)
    {
        CAN_Receive(tCAN, 0, &rrMsg);
        u8CAN_PackageFlag=1;
    }
}
```

3.2.2 下位机

3.2.2.1 下位机主程序

若需要一对多的沟通方式，下位机须要定义新的device的ID，如：#define Device1_ISP_ID 0x803。

```
#define PLL_CLOCK                       48000000
#define CAN_BAUD_RATE                   250000
#define Master_ISP_ID                   0x800
#define Device0_ISP_ID                  0x802
#define CAN_ISP_DtatLength              0x08
#define CAN_FailTime                    0xff
int main(void)
{
    /* Unlock protected registers */
    SYS_UnlockReg();
    /* Init System, IP clock and multi-function I/O */
    SYS_Init();
    /* Init CAN port */
    can_init();
    /* Enable FMC ISP function */
    FMC_Open();
    if(APROM_detect()>0)
    {
        /* stat update program */
        if(CAN_ISP_Update(CAN0)>0)
        {
            while(u8CAN_AckFlag);
            goto _APROM;
        }
    }

    while(1);
    _APROM:
    /*clear bit*/
    outpw(&SYS->RSTSTS, 3);
    outpw(&FMC->ISPCTL, inpw(&FMC->ISPCTL) & 0xFFFFFFFC);
    outpw(&SCB->AIRCR, (V6M_AIRCR_VECTKEY_DATA | V6M_AIRCR_SYSRESETREQ));
    /* Trap the CPU */
    while(1);
}
```

3.2.2.2 下位机ISP更新程序

```
uint8_t CAN_ISP_Update(CAN_T *tCAN)
{
    STR_CANMSG_ISP CAN_buffer;
    uint32_t APROM_Data=0;
    uint8_t Verify_flag=1;
    memset(&CAN_buffer, 0, sizeof(CAN_buffer));
    FMC_EnableAPUpdate();

    while(1)
    {
        u32CANWaitCount = 0;
        u8CAN_PackageFlag=0;
        Verify_flag=1;
        CLK_EnableSysTickINT(10000);                        // System Tick timer value us
        while(u8CAN_PackageFlag == 0 && TimeOut_flag == 0); //receive CAN ISP Package
        if(TimeOut_flag)
        {
            CLK_DisableSysTick();
            FMC_DisableAPUpdate();
            return FALSE;
        }
        CLK_DisableSysTick();
        memcpy(&CAN_buffer.Address, &rrMsg.Data, 4);       //update address
        if(((CAN_buffer.Address % FMC_FLASH_PAGE_SIZE) == 0) || (CAN_buffer.Address == 0))
        {
            FMC_Erase(CAN_buffer.Address);
            if(VerifyData(CAN_buffer.Address, CAN_buffer.Address + FMC_FLASH_PAGE_SIZE,
0xFFFFFFFF) < 0) Verify_flag=0;
        }
        if(Verify_flag)
        {
            memcpy(&APROM_Data, &rrMsg.Data[4], 4);        //update data
            FMC_Write(CAN_buffer.Address, APROM_Data);     //program ROM
            u8CAN_AckFlag = 1;
            CAN_Package_ACK(CAN0);                         //send CAN ISP Package (ACK)
            if(CAN_buffer.Address == Chip_EndAddress)      //check CAN ISP EndCode
            {
                FMC_DisableAPUpdate();
                return TRUE;
            }
```

```
        }
    }
}
```

3.2.2.3 下位机传送、接收Message

若需要一对多的沟通方式，下位机须要增加device传送给master message的设定，是哪一个device的数据要透过哪一个报文的信道传送数据，报文信道可设定范围为0至31，共32个信道，如程序中批注二及批注三所示。

```
void CAN_Package_ACK(CAN_T *tCAN)
{
    STR_CANMSG_T tMsg;
    /* Send a 11-bit Standard Identifier message */
    tMsg.FrameType = CAN_DATA_FRAME;
    tMsg.IdType    = CAN_STD_ID;
    /* Setting Device ID */ 批注二
    tMsg.Id        = Device0_ISP_ID;
    tMsg.DLC       = CAN_ISP_DtatLength;
    memcpy(&tMsg.Data, &rrMsg.Data, 8);
    /* Configure Msg RAM and send the Msg in the RAM*/ 批注三
    if(CAN_Transmit(tCAN, MSG(5), &tMsg) == FALSE)
    {
        return;
    }
}

uint8_t can_setRxMsg(CAN_T *tCAN)
{
    /* Receiving the data of the Device0_ISP_ID to channel0 of MSG SRAM */
    if(CAN_SetRxMsg(tCAN, MSG(0), CAN_STD_ID, Master_ISP_ID) == FALSE)
    {
        return FALSE;
    }
    return TRUE;
}
```

3.2.2.4 下位机中断程序

```
void CAN0_IRQHandler(void)
{
    uint32_t u8IIDRstatus;
    u8IIDRstatus = CAN0->IIDR;
    /* Check Status Interrupt Flag (Error status Int and Status change Int) */
    if(u8IIDRstatus == 0x00008000)
    {
        /*************************/
        /* Status Change interrupt*/
        /*************************/
        if(CAN0->STATUS & CAN_STATUS_RXOK_Msk)
        {
            /* Clear RxOK status*/
            CAN0->STATUS &= ~CAN_STATUS_RXOK_Msk;
        }

        if(CAN0->STATUS & CAN_STATUS_TXOK_Msk)
        {
            /* Clear TxOK status*/
            CAN0->STATUS &= ~CAN_STATUS_TXOK_Msk;
            u8CAN_AckFlag=0;
        }
    }
    else if(u8IIDRstatus!=0)
    {
        CAN_MsgInterrupt(CAN0, u8IIDRstatus);
        /* Clear Interrupt Pending */
        CAN_CLR_INT_PENDING_BIT(CAN0, (u8IIDRstatus - 1));
    }

void CAN_MsgInterrupt(CAN_T *tCAN, uint32_t u32IIDR)
{
    if(u32IIDR == 1)
    {
        CAN_Receive(tCAN, 0, &rrMsg);
        u8CAN_PackageFlag=1;
    }
}
```

4 结论

科技进步的时代许多控制都已电子化或加上传感器，为了能缩减通讯线缆的使用量，不管是车用还是航空、航海电子仪器、工厂自动化、电梯、医疗仪器都已采用 CAN Bus作为通讯方式，可以透过CAN ISP更新开发时尚未发现的疏失、以及达到产品功能完整性，来延长产品的生命周期。

若航空、航海电子仪器、工厂自动化、电梯 进行维护更新必须透过计算机更新是很不方便的。本文针此困扰，提供创新便利的更新模式。这些产品的传感器信息、控制讯号透过CAN Bus传输数据，本文藉由CAN Bus通讯方式作为更新韧体的接口，所以产品外部只须预留单一个CAN Bus通讯接口即可进行系统维护。目前本文CAN ISP 更新韧体的应用也已使用在汽车感测系统。

5 Revision History

Date	Revision	Description
2018.08.06	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.