

使用KE06上的四线式I²C接口

作者: Ben Wang

1 简介

本应用笔记将描述使用Kinetis E系列KE06上的四线式I²C接口的流程。本应用笔记中提供的示例代码在KE06上经过测试，测试方法为在两个评估板之间进行I²C主/从通信。

目录

1	简介.....	1
2	KE06 I ² C 概述.....	2
3	四线式接口特性.....	2
3.1	四线式接口配置.....	2
4	四线式接口测试.....	3
4.1	硬件设置.....	3
4.2	软件设置.....	5
5	参考.....	7
6	修订历史记录.....	7

2 KE06 I²C概述

KE06包含两个支持SMBus功能的I²C模块。I²C接口提供了一种多个器件之间通信的方式。该接口在总线负载和时序最高的情况下以高达100 kb/s的速度运行。当减小总线负载时，器件能够以更高的波特率运行，可以达到总线频率的1/20。最大通信长度和可连接的器件数量受400 pF的最大总线电容限制。

此外，KE06 I²C0还提供四线式接口选项。

3 四线式接口特性

大多数设备应用中都会存在很大的总线负载和大量的总线噪声。为了提供稳健的I²C通信，可能需要额外的总线开关或线路驱动器。对于传统的两线式I²C通信，这将会增加额外的系统物料(BOM)成本。传统的I²C接口采用SDA/SCL双向传输，四线式接口则增加了一个选项，将其分为单向的输入和输出功能。在四个引脚的配置中，SDA_IN、SDA_OUT、SCL_IN和SCL_OUT引脚均带有反相输出。该配置可使用户在设计线路驱动器时的额外成本最低，并且可用于改善I²C总线的抗噪声能力。该配置可以通过增加两个晶体管、六个电阻和两个二极管来实现。

3.1 四线式接口配置

当KE06的SIM_SOPT1[I2C04WEN]位被置位时，四线式接口使能。SDA_IN/SCL_IN (KE06上的PTA2/PTA3) 管脚用来表示SDA/SCL信号的输入，SDA_OUT/SCL_OUT (KE06上的PTA1/PTA0) 管脚用来表示SDA/SCL信号的输出。

通过置位SIM_SOPT1[I2C04WEN]位而使能四线式接口功能后，用户可将SIM_SOPT1[I2C00INV]位置位，使SDA_OUT/SCL_OUT在输出前反相。

但是，该功能仅当未重新映射I²C0管脚时可用。

下面为I²C0四线式接口的示意图：

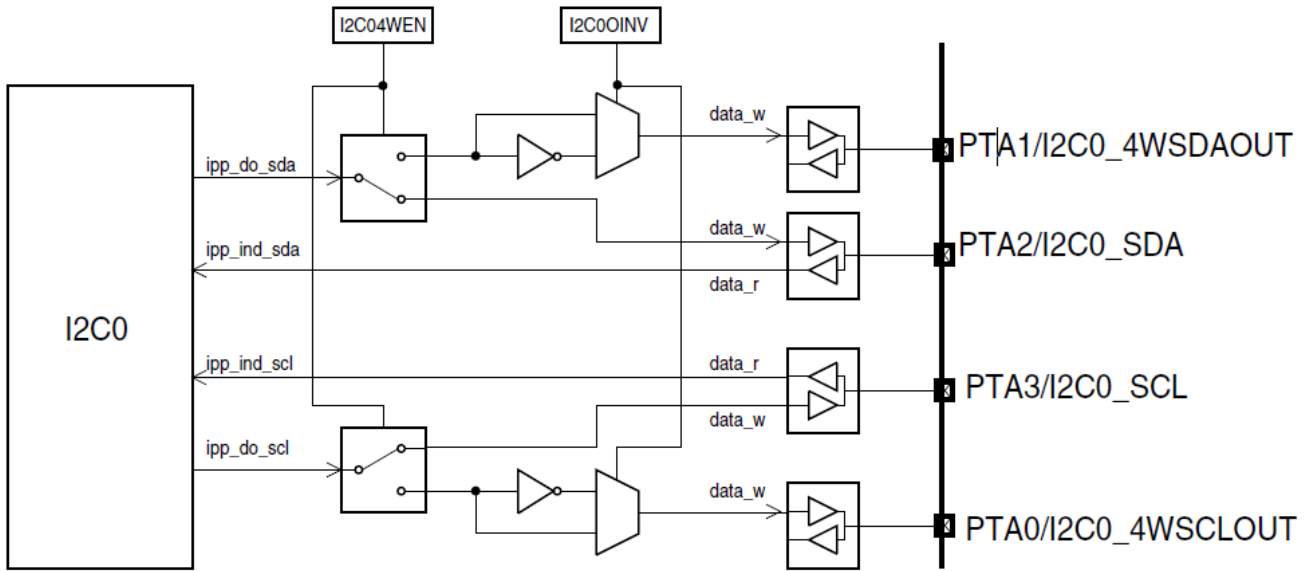


图1. I²C0四线式接口图

4 四线式接口测试

四线式接口采用内部评估板进行测试。如前所述，采用四线式I²C无需对I²C的外设重新设计，仅需换一个接口即可。

4.1 硬件设置

图1表示通用的四线式I²C接口连接。

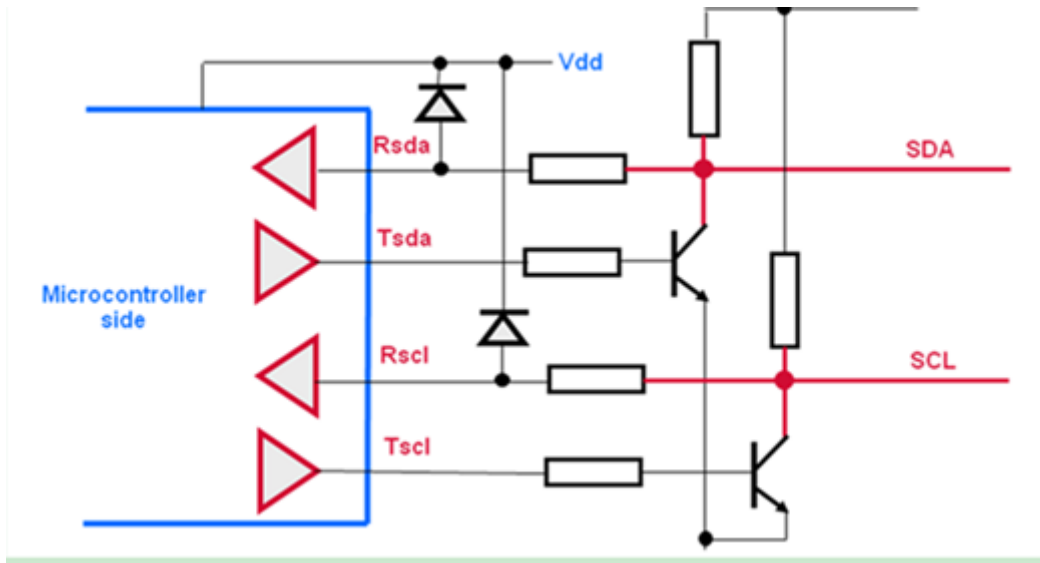


图2. 四线式I²C接口连接

下图表示评估板上实际的四线式I²C连接:

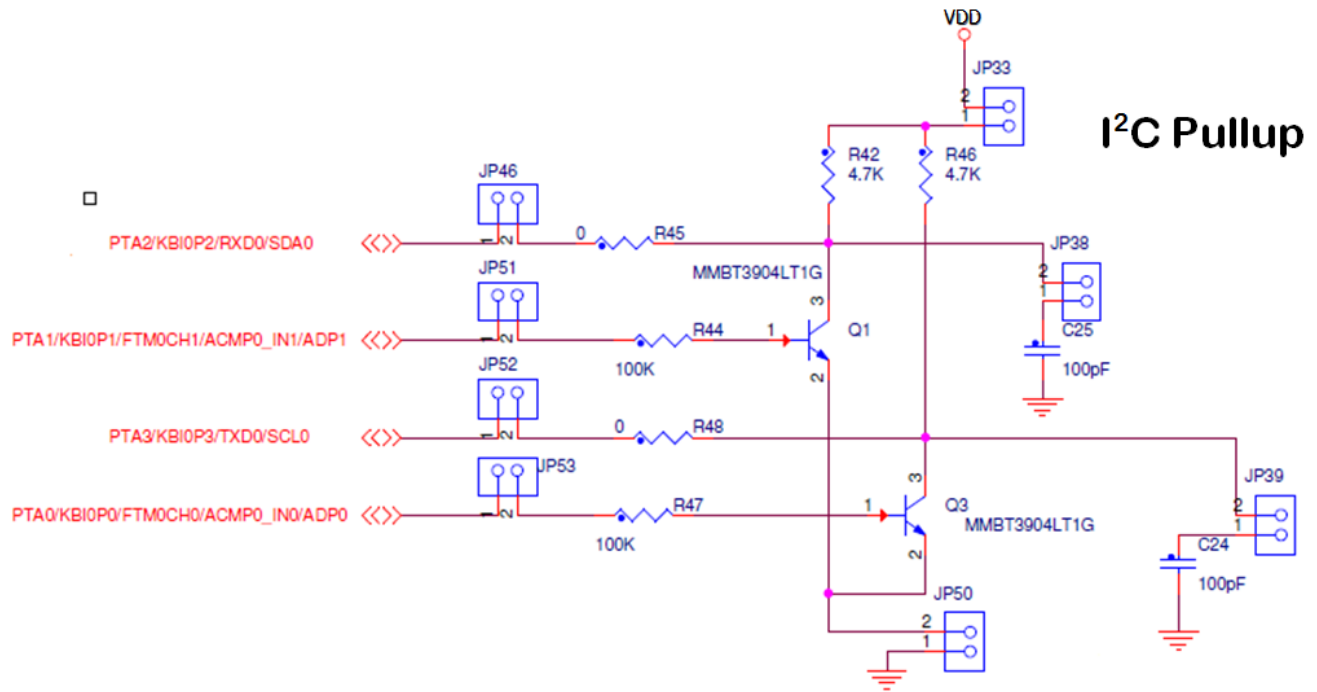


图3. I²C0四线式硬件原理图

4.2 软件设置

在对四线式接口进行测试时，仅SIM_SOPT1寄存器中的两个位需要在软件中置位。其他I²C主/从配置与两线式相同。

I²C四线式通信的主机代码如下：

I2C_4wire_master.c:

```
int main (void)
{
    I2C_ConfigType  sI2C_Config = {0};
    uint32_t i,j;

    for(i=0;i<0xffff;i++);

    printf("\nRunning the I2C_4wire_master project.\n");
    printf("\nThis test should be run on EVB.\n");

    UART_WaitTxComplete(TERM_PORT);

    /* Initialize I2C module with poll mode */
    sI2C_Config.u16F = 0x1F;
    sI2C_Config.sSetting.bIntEn = 0;
    sI2C_Config.sSetting.bI2CEn = 1;

    I2C_Init(I2C0,&sI2C_Config );
#ifdef I2C0_4WIRE_ENABLE
    SIM_Enable4WireI2C0();
#endif

#ifdef I2C0_4WIRE_OUT_INVERT
    SIM_EnableI2C0OuputInvertion();
#endif

    for(i=0;i<64;i++)
    {
        u8I2C_SendBuff[i] = i;
    }

    while(1)
    {
#ifdef I2C0_4WIRE_ENABLE
        printf("\nPress any key to make Master Send data to slave with four-wire mode.\n");
#else
        printf("\nPress any key to make Master Send data to slave with 2-wire mode.\n");
#endif

        I2C_MasterSendWait(I2C0,I2C_SLAVE_ADDRESS1,&u8I2C_SendBuff[0],64);
        I2C_MasterReadWait(I2C0,I2C_SLAVE_ADDRESS1,&u8I2C_ReceiveBuff[0],64);
        printf("Read data from I2C slave:\n");
        for(i=0;i<8;i++)
        {
            for(j=0;j<8;j++)
            {
                printf("0x%x,", u8I2C_ReceiveBuff[i*8+j]);
            }
        }
    }
}
```

```

    }
    printf("\n");
}
for(i=0;i<64;i++)
{
    u8I2C_SendBuff[i] += i;
}
for(i=0;i<0xfffff;i++);
}
}

```

I²C四线式通信的从设备代码如下:

I2C_4wire_slave.c:

```

int main (void)
{
    uint8_t      u8I2C_ReceiveLength;
    uint32_t     i;

    I2C_ConfigType  sI2C_Config = {0};

    printf("\nRunning the I2C_4wire_slave project.\n");

    UART_WaitTxComplete(TERM_PORT);

    /* initialize I2C0 global variable and call back function*/
    I2C0_InitGlobalVariable( );

    sI2C_Config.ul6Slt = 0;
    sI2C_Config.ul6F = 0x1F;
    sI2C_Config.ul6OwnA1 = I2C_SLAVE_ADDRESS;
    sI2C_Config.sSetting.bIntEn = 1;
    sI2C_Config.sSetting.bI2CEn = 1;

    I2C_Init(I2C0,&sI2C_Config);

#ifdef I2C0_4WIRE_ENABLE
    SIM_Enable4WireI2C0();
#endif

#ifdef I2C0_4WIRE_OUT_INVERT
    SIM_EnableI2C0OuputInverction();
#endif

    for(i=0;i<64;i++)
    {
        u8I2C_SendBuff[i] = i;
    }
    u8I2C_SendBuff[0] = 0xa0;
    I2C0_SlaveSend(u8I2C_SendBuff,64);
    while(1)
    {

```

```

u8I2C_ReceiveLength = I2C0_SlaveReceive(&u8I2C_ReceiveBuff[0]);

I2C0_SlaveSend(&u8I2C_ReceiveBuff[0], 64);

if( u8I2C_ReceiveLength )
{
    printf("I2C received data:\n");
    for(i=0;i<u8I2C_ReceiveLength;i++)
    {
        if( (i%8) == 0 )
        {
            printf("\n");
        }
        printf("0x%x,", u8I2C_ReceiveBuff[i]);
    }
    printf("\n");
}
for(i=0;i<0xfffff;i++);
}
}

```

5 参考

可从freescale.com获取以下参考文档:

- KE06参考手册
- KE06数据手册

6 修订历史记录

修订版本号	日期	重要改动
0	2014/03	初始版本

How to Reach Us:

Home Page:

freescale.com

Web Support:

freescale.com/support

本文档中的信息仅供系统和软件实施方使用 Freescale 产品。本文并未明示或者暗示授予利用本文档信息进行设计或者加工集成电路的版权许可。Freescale 保留对此处任何产品进行更改的权利，恕不另行通知。

Freescale 对其产品在任何特定用途方面的适用性不做任何担保、表示或保证，也不承担因为应用程序或者使用产品或电路所产生的任何责任，明确拒绝承担包括但不限于后果性的或附带性的损害在内的所有责任。Freescale 的数据表和 / 或规格中所提供的 "典型" 参数在不同应用中可能并且确实不同，实际性能会随时间而有所变化。所有运行参数，包括 "经典值" 在内，必须经由客户的技术专家对每个客户的应用程序进行验证。Freescale 未转让与其专利权及其他权利相关的许可。Freescale 销售产品时遵循以下网址中包含的标准销售条款和条件：freescale.com/SalesTermsandConditions。

Freescale, the Freescale logo are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners.

© 2014 Freescale Semiconductor, Inc.

© 2014 飞思卡尔半导体有限公司。

