Mini51DE I2C LIRC Master

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	This document describes how to set I2C clock source to LIRC and how a Master access Slave.
BSP Version	Mini51DE Series BSP CMSIS V3.02.000
Hardware	NuTiny-EVB-Mini51_V2.1

1 Function Description

1.1 Introduction

This example code demonstrates how to set clock source of I2C to LIRC and how a Master access Slave. The system clock source is LIRC. Since clock source of Mini51 I2C is the same as system clock. Hence, the clock source of I2C is also from LIRC. The bus clock of I2C is set to 833Hz. Then, Master accesses Slave with Byte Write and Byte Read operations, and check if the read data is equal to the programmed data.

1.2 Demo Result

We compile the example code and download it to the NuTiny-EVB-Mini51_V2.1 development board. The bus clock of I2C is set to 833Hz. Then, Master accesses Slave with Byte Write and Byte Read operations, and check if the read data is equal to the programmed data, as shown in Figure 1.

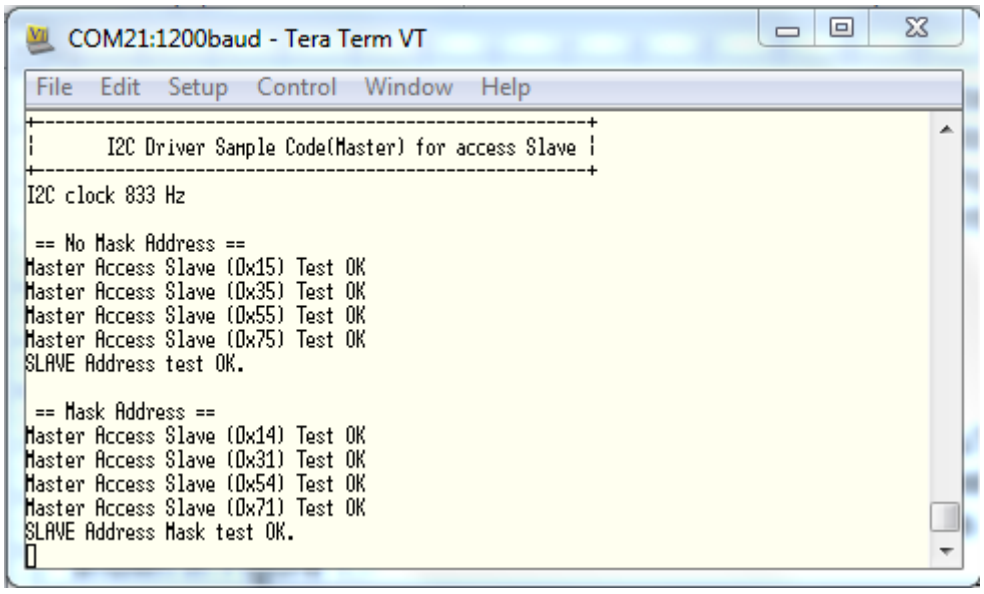


Figure 1 Master access Slave

2 Code Description

In this example, we have to configure HCLK clock source to LIRC. Enable UART and I2C clock. Following is the related code segment in SYS_Init() of main.c

```
    /* Switch HCLK clock source to LIRC */
    CLK_SetHCLK(CLK_CLKSEL0_HCLK_S_LIRC,CLK_CLKDIV_HCLK(1));

    /* Enable IP clock */
    CLK_EnableModuleClock(UART_MODULE);
    CLK_EnableModuleClock(I2C_MODULE);
```

Besides, set UART and I2C multi-function pin in SYS_Init() of main.c

```
    /* Set P0 multi-function pins for UART RXD and TXD */
    SYS->P0_MFP &= ~(SYS_MFP_P01_Msk | SYS_MFP_P00_Msk);
    SYS->P0_MFP |= (SYS_MFP_P01_RXD | SYS_MFP_P00_TXD);

    /* Set P3.4 and P3.5 for I2C SDA and SCL */
    SYS->P3_MFP = SYS_MFP_P34_SDA | SYS_MFP_P35_SCL;
```

I2C_Init() initializes I2C clock setting. Set four slave addresses. Enable I2C interrupt and NVIC.

```
void I2C_Init(void)
{
    /* Open I2C and set clock to 833 Hz */
    I2C_Open(I2C, 833);

    /* Get I2C Bus Clock */
    printf("I2C clock %d Hz\n", I2C_GetBusClockFreq(I2C));

    /* Set I2C 4 Slave Addresses */
    I2C_SetSlaveAddr(I2C, 0, 0x15, 0);   /* Slave Address : 0x15 */
    I2C_SetSlaveAddr(I2C, 1, 0x35, 0);   /* Slave Address : 0x35 */
    I2C_SetSlaveAddr(I2C, 2, 0x55, 0);   /* Slave Address : 0x55 */
    I2C_SetSlaveAddr(I2C, 3, 0x75, 0);   /* Slave Address : 0x75 */

    I2C_EnableInt(I2C);
    NVIC_EnableIRQ(I2C_IRQn);
}
```

The interrupt service routine, I2C_IRQHandler(), gets interrupt status and calls the related handler.

```
void I2C_IRQHandler(void)
{
    uint32_t u32Status;

    u32Status = I2C_GET_STATUS(I2C);

    if (I2C_GET_TIMEOUT_FLAG(I2C))
    {
        /* Clear I2C Timeout Flag */
        I2C_ClearTimeoutFlag(I2C);
    }
    else
    {
        if (s_I2CHandlerFn != NULL)
            s_I2CHandlerFn(u32Status);
    }
}
```

In main routine, initializes UART baud rate to 1200.

```
   /* Init UART to 1200-8n1 for print message */
    SYS_ResetModule(UART_RST);
    UART_Open(UART0, 1200);
```

Besides, calls I2C_Init() for initializing I2C. Then start to access slave.

```
    /* Init I2C */
    I2C_Init();

    /* Access Slave with no address mask */
    printf("\n");
    printf(" == No Mask Address ==\n");
    Read_Write_SLAVE(0x15);
    Read_Write_SLAVE(0x35);
    Read_Write_SLAVE(0x55);
    Read_Write_SLAVE(0x75);
    printf("SLAVE Address test OK.\n");

    /* Access Slave with address mask */
    printf("\n");
    printf(" == No Mask Address ==\n");
    Read_Write_SLAVE(0x15);
    Read_Write_SLAVE(0x35);
```

```
    Read_Write_SLAVE(0x55);
    Read_Write_SLAVE(0x75);
    printf("SLAVE Address test OK.\n");
}
```

3 Software and Hardware Environment

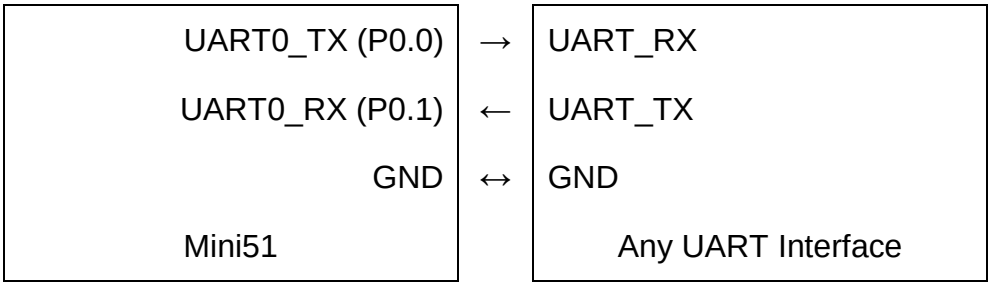
Software Environment

- BSP version
  - Mini51DE Series BSP CMSIS V3.02.000
- IDE version
  - Keil uVersion 5.26

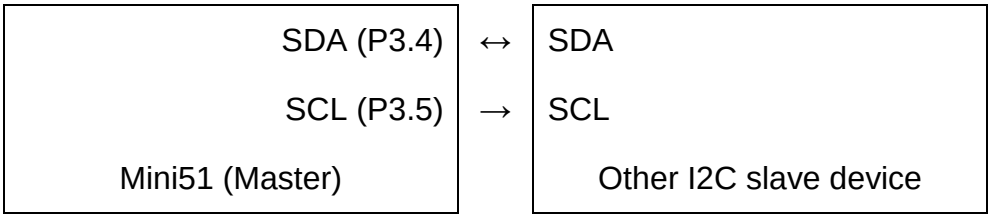
Hardware Environment

- Circuit components
  - NuTiny-EVB-Mini51_V2.1
- Diagram

We use printf function to output message while accessing slave. The UART0 TX RX pins are P0.0 and P0.1 respectively.



Set I2C SDA and SCL pins to P3.4 and P3.5.



4 Directory Information

EC_Mini51_I2C_LIRC_Master_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

5 How to Execute Example Code

1. Browsing into sample code folder by section 4 Directory Information and double click I2C_LIRC_Master.uvproj.
2. Enter Keil compile mode
   a. Build
   b. Download
   c. Start/Stop debug session
3. Enter debug mode
   a. Run

6 Revision History

Date	Revision	Description
Jun. 17, 2019	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.