读取传感器DHT11的湿度和温度数据

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

代码简述	使用NUC240的GPIO (PA.0)和TIMER0读取DHT11的湿度和温度数据
BSP 版本	NUC230/240 Series BSP CMSIS V3.01.001
开发平台	NuTiny-SDK-NUC240V V1.2

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 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 功能介绍

1.1 简介

使用GPIO (PA.0)和TIMER0读取DHT11的湿度和温度数据。DHT11是个常见的数字湿度、温度传感器。DHT11使用简化的单总线通讯方式交换数据。数据长度为40位，数据格式如所下表所示：

1st 8 比特	2nd 8 比特	3rd 8 比特	4th 8 比特	5th 8 比特
湿度整数数据	湿度小数数据	温度整数数据	温度小数数据	前4笔资料的校验码

藉由GPIO和TIMER0，使用者可以产生与DHT11通讯的波形。请从以下链接，参考DHT11的规格数据，了解更详细的时序图。
https://osepp.com/downloads/pdf/DHT11-Technical-Data-Sheet-Translated-Version.pdf

1.2 执行结果

每1.5秒读取DHT11的湿度和温度数据。结果如下图所示：

```
CPU @ 71884800 Hz
+------------------------------------------------+
|    Read DHT11 data Sample Code                 |
|    Use GPIO(PA.0) and TIMER0 to read DHT11 data|
|    Please connect PA.0 to DHT11 data pin       |
+------------------------------------------------+

Humidity: 72.0, Temperature: 24.9
Humidity: 73.0, Temperature: 24.9
Humidity: 73.0, Temperature: 24.9
Humidity: 73.0, Temperature: 24.9
Humidity: 73.0, Temperature: 24.9
```

2 代码介绍

发送开始讯号给DHT11，使DHT11从低功耗模式切换到执行模式。

```
    /* The Quasi-bidirection mode could be used as input with pull-up enable */
    GPIO_SetMode(Data_Port, Data_PinMsk, GPIO_PMD_QUASI);

    /* Send out start signal to DHT11 */
    /* Set data pin level from high to low */
    Data_Pin = 1;
    Data_Pin = 0;
    /* Take at least 18 ms to ensure DHT11's detection of signal */
    CLK_SysTickDelay(18000);
    /* Pull up data pin */
    Data_Pin = 1;

    /* Record current timer counter value */
    u32_poll_time = TIMER_GetCounter(TIMER0);

    /* Wait response from DHT11 by pulling down data pin */
    while (1)
    {
        if (Data_Pin == 0)
        {
            break;
        }

        if ((TIMER_GetCounter(TIMER0) - u32_poll_time) > 1000)
            /* DHT11 doesn't response over 1 ms */
            return FALSE;
    }

    /* Wait until DHT11 pulls up data pin level */
    while (1)
    {
        if (Data_Pin == 1)
        {
            break;
        }
    }
```

```
    /* Send start signal successfully */
    return TRUE;
```

在NUC240送出开始讯号并且收到DHT11的回复后，NUC240开始读取DHT11的数据。

```
    /* Init TIMER0 to measure high level time */
    TMR0_Init();

    /* Start communicate with DHT11 */
    if (DHT11_Reset() == TRUE)
    {
        /* Wait until DHT11 starts data transmission */
        while (Data_Pin == 1);

        /* A complete data transmission is 40 bits, DHT11 sends MSB first */
        /* 1st 8 bits is humidity integral data */
        g_u8Humidity_Int = 0;

        for (i = 0; i < 8; i++)
        {
            /* Shift current data for next bit */
            g_u8Humidity_Int = g_u8Humidity_Int << 1;

            /* Wait until DHT11 pull high data pin */
            while (Data_Pin == 0);

            /* Measure the high level time to determine the data bit */
            u32_prev_time = TIMER_GetCounter(TIMER0);

            while (Data_Pin == 1);

            if ((TIMER_GetCounter(TIMER0) - u32_prev_time) > Max_Data0_Time)
            {
                g_u8Humidity_Int |= 0x01;
            }
        }

        /* 2nd 8 bits is humidity decimal data */
        g_u8Humidity_Dec = 0;

        for (i = 0; i < 8; i++)
        {
```

```
            /* Shift current data for next bit */
            g_u8Humidity_Dec = g_u8Humidity_Dec << 1;

            /* Wait until DHT11 pull high data pin */
            while (Data_Pin == 0);

            /* Measure the high level time to determine the data bit */
            u32_prev_time = TIMER_GetCounter(TIMER0);

            while (Data_Pin == 1);

            if ((TIMER_GetCounter(TIMER0) - u32_prev_time) > Max_Data0_Time)
            {
                g_u8Humidity_Dec |= 0x01;
            }
        }

        /* 3rd 8 bits is temperature integral data */
        g_u8Temperature_Int = 0;

        for (i = 0; i < 8; i++)
        {
            /* Shift current data for next bit */
            g_u8Temperature_Int = g_u8Temperature_Int << 1;

            /* Wait until DHT11 pull high data pin */
            while (Data_Pin == 0);

            /* Measure the high level time to determine the data bit */
            u32_prev_time = TIMER_GetCounter(TIMER0);

            while (Data_Pin == 1);

            if ((TIMER_GetCounter(TIMER0) - u32_prev_time) > Max_Data0_Time)
            {
                g_u8Temperature_Int |= 0x01;
            }
        }

        /* 4th 8 bits is temperature decimal data */
        g_u8Temperature_Dec = 0;
```

```
    for (i = 0; i < 8; i++)
    {
        /* Shift current data for next bit */
        g_u8Temperature_Dec = g_u8Temperature_Dec << 1;

        /* Wait until DHT11 pull high data pin */
        while (Data_Pin == 0);

        /* Measure the high level time to determine the data bit */
        u32_prev_time = TIMER_GetCounter(TIMER0);

        while (Data_Pin == 1);

        if ((TIMER_GetCounter(TIMER0) - u32_prev_time) > Max_Data0_Time)
        {
            g_u8Temperature_Dec |= 0x01;
        }
    }

    /* 5th 8 bits is checksum */
    u8Checksum = 0;

    for (i = 0; i < 8; i++)
    {
        /* Shift current data for next bit */
        u8Checksum = u8Checksum << 1;

        /* Wait until DHT11 pull high data pin */
        while (Data_Pin == 0);

        /* Measure the high level time to determine the data bit */
        u32_prev_time = TIMER_GetCounter(TIMER0);

        while (Data_Pin == 1);

        if ((TIMER_GetCounter(TIMER0) - u32_prev_time) > Max_Data0_Time)
        {
            u8Checksum |= 0x01;
        }
    }
```

```
        /* Stop TIMER0 */
        TIMER_Stop(TIMER0);

        /* Compare received data and checksum */
        if ((uint8_t)(g_u8Humidity_Int + g_u8Humidity_Dec + g_u8Temperature_Int +
g_u8Temperature_Dec) == u8Checksum)
        {
            return TRUE;
        }
        else
        {
            printf("Checksum error!\n");

            return FALSE;
        }
    }
    else
    {
        /* Stop TIMER0 */
        TIMER_Stop(TIMER0);

        printf("DHT11 is no response.\n");

        return FALSE;
    }
```

NUC240每隔1.5秒读取DHT11的湿度和温度数据。

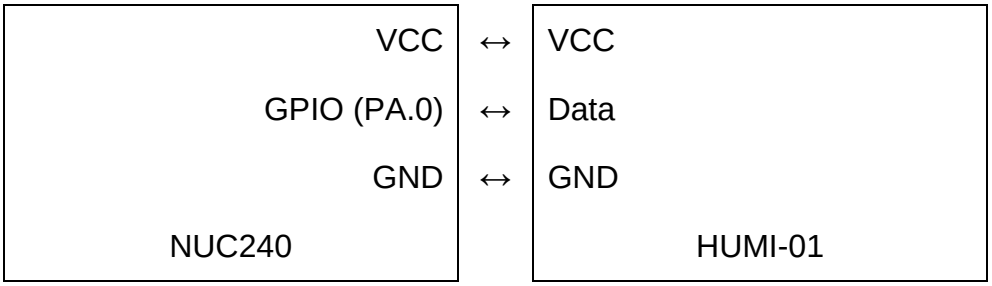
```
while (1)
    {
        /* Delay 1.5 seconds */
        Delay_mS(1500);

        if (DHT11_Read() == TRUE)
        {
            printf("Humidity: %d.%d, ", g_u8Humidity_Int, g_u8Humidity_Dec);
            printf("Temperature: %d.%d\n", g_u8Temperature_Int, g_u8Temperature_Dec);
        }
    }
```

3 软件与硬件环境

- 软件环境
  - BSP 版本
    - NUC230/240 Series BSP CMSIS V3.01.001
  - IDE 版本
    - Keil uVersion 5.26
- 硬件环境
  - 电路组件
    - NuTiny-SDK-NUC240V V1.2
    - 湿度和温度传感器 DHT11 (模块 HUMI-01)
  - 示意图

NUC240使用GPIO (PA.0)连接模块HUMI-01的Data脚位。



4 目录信息

- EC_NUC240_Read_DHT11_Data_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 如何执行范例程序

1. 根据目录信息章节进入 ExampleCode 路径中的 KEIL 文件夹，双击 NUC240_Read_DHT11_Data.uvproj。
2. 进入编译模式接口
   a. 编译
   b. 下载代码至内存
   c. 进入 / 离开除错模式
3. 进入除错模式接口
   a. 执行代码

6 修订纪录

Date	Revision	Description
June 26, 2019	1.00	1. 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.