Use Nano100 to read DS18B20

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	This code uses Nano100 to read DS18B20
BSP Version	Nano100B Series BSP CMSIS v3.03.000
Hardware	NuTiny-EVB-Nano130-LQFP128 v1.1

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

www.nuvoton.com

1 Function Description

1.1 Introduction

The MCU communicates with DS18B20 through PD.0, reads DS18B20 and then outputs the temperature through serial port as float type. DS18B20 temperature resolution is 12 bits. Serial port baud rate is 115200.

1.2 Principle

The UART0 TX pin of MCU is PB.1.
One wire bus protocol, you can refer to the DS18B20 Datasheet.

1.3 Demo Result

```
+-------------------------------------+
|    Temperature Sensor Sample Code   |
+-------------------------------------+
Temperature raw data = 25.062500
Temperature raw data = 25.062500
Temperature raw data = 25.062500
Temperature raw data = 25.062500
Temperature raw data = 25.125000
Temperature raw data = 25.312500
Temperature raw data = 25.500000
Temperature raw data = 25.812500
Temperature raw data = 26.062500
Temperature raw data = 26.312500
Temperature raw data = 26.625000
Temperature raw data = 26.812500
Temperature raw data = 27.062500
Temperature raw data = 27.250000
```

2 Code Description

Reading the temperature of the DS18B20 is simple. There are two stages, and each stage has three steps. The first stage, sending reset, sending skip ROM, and sending convert temperature commands in order. The second stage, sending reset, sending skip ROM, and sending read temperature commands in order.

Note that initializing the DS18B20 is resetting it.

```
            /* Step 1: Initial DS18B20 */
            if (init_DS18B20())
            {
                /* Step 2: ROM Command */
                writeDS18B20_byte(0xCC);    /* Skip ROM command */
                /* Step 3: Function Command */
                writeDS18B20_byte(0x44);  /* Convert temperature */

                /* Delay for converting temperature */
                TIMER_Delay(TIMER0, 1000000);

                /* Step 1: Initial DS18B20 */
                if (init_DS18B20())
                {
                    /* Step 2: ROM Command */
                    writeDS18B20_byte(0xCC);    /* Skip ROM command */
                    /* Step 3: Function Command */
                    writeDS18B20_byte(0xBE);    /* Read temperature data*/
                    /* Get low byte data of temperature*/
                    g_u8DataL = readDS18B20_byte();
                    /* Get high byte data of temperature*/
                    g_u8DataH = (readDS18B20_byte() & 0x0f);
                    /* Combine high byte data with low byte data */
                    g_u16Data = (g_u8DataH << 8) | g_u8DataL;

                    printf("Temperature raw data = %f\n", DataCoding(g_u16Data));
                }
            }
```

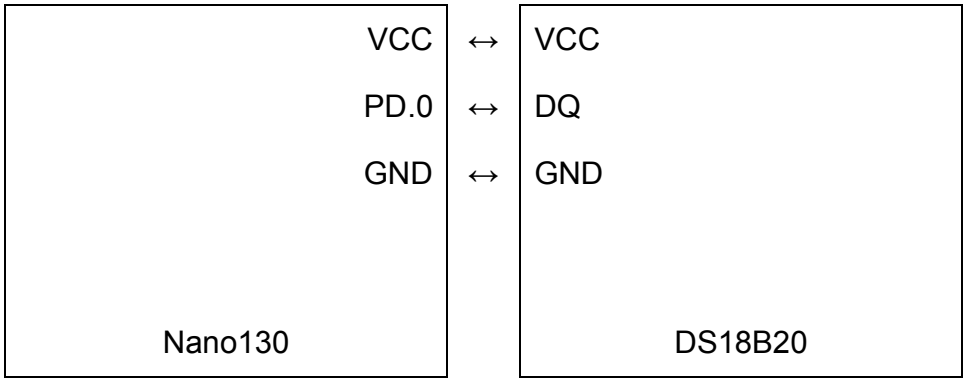
3 Software and Hardware Environment

Software Environment

- BSP version
  - Nano100B Series BSP CMSIS v3.03.000
- IDE version
  - Keil uVersion 5.26

Hardware Environment

- Circuit components
  - NuTiny-EVB-Nano130-LQFP128 v1.1
  - DS18B20
- Diagram



4 Directory Information

- EC_Nano100_GPIO_1Wire_Ds18b20_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 Directory Information (section 4) and double click Nano100_GPIO_1Wire_Ds18b20.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
Aug. 19, 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.