L-C Sensor

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	L-C Sensor is a device for inductive sensing the rotation of a mechanical wheel based on a circuit containing both an inductor and a capacitor.
BSP Version	Nano100B_Series_BSP_CMSIS_v3.02.002
Hardware	NuTiny-Nano100K

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 Function Description

1.1 Introduction

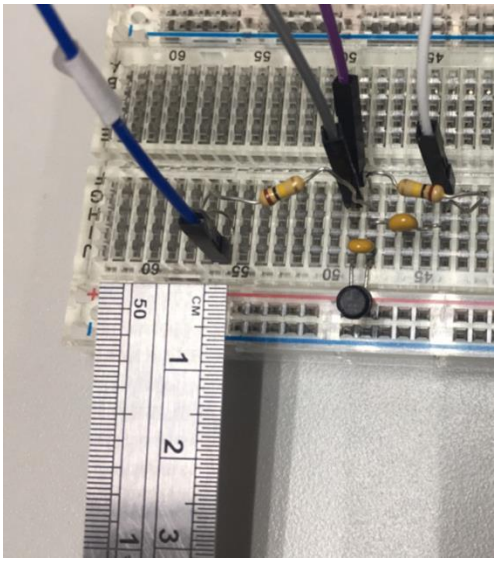
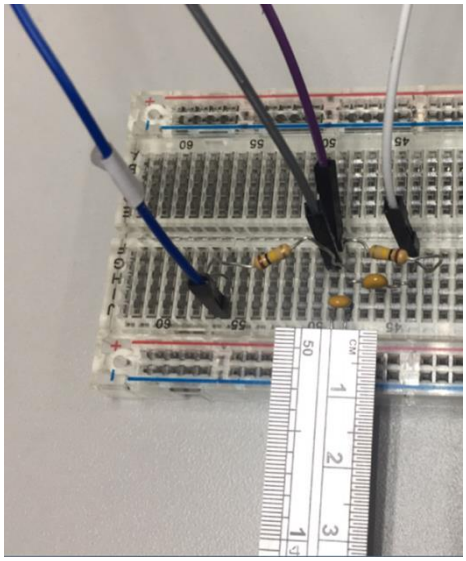
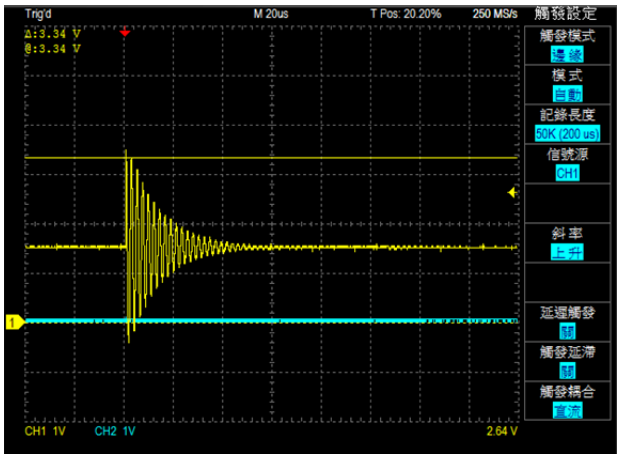
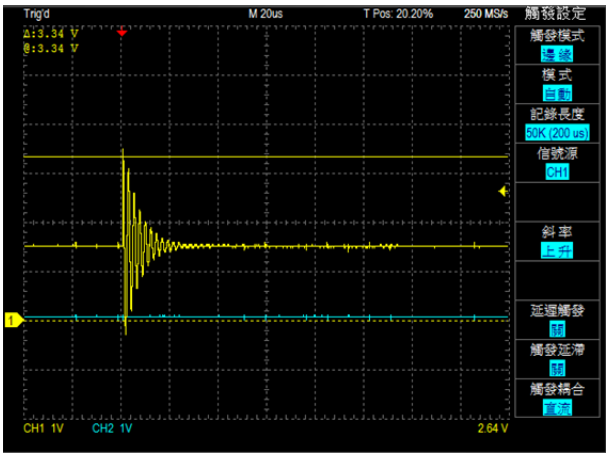
L-C Sensor is a device for inductive sensing the rotation of a mechanical wheel. The L-C sensor based on a circuit containing both an inductor (L) and a capacitor (C) can oscillate without a source of emf by shifting the energy stored in the circuit between the electric and magnetic fields.

1.2 Principle

When L-C resonant circuit is charged, it will generate a resonant signal at a fixed frequency. This signal will decay to zero when time passes. If there is some ferromagnetic material coming close to the coil inductance, the amplitude of resonant signal will decay faster. Set a threshold for determining how many times the resonant signal is over the threshold. By this result, user can check if there is a ferromagnetic material coming close to the coil inductance or not. In this schematic In this sample code, the L-C resonant circuit will bias on $V_{in}/2$. Because the maximum of ADC conversion value is 4095(0xFFF), the bias on $V_{in}/2$ will convert to 2047(0x7FF). Hence it is recommended that the threshold value be set on 0x850~0x8A0. User can choose the suitable threshold value on this range.
On this article, the L-C resonant circuit consists of a 220 pF capacitor and a 470 uH inductance. The resonant frequency is 494,948 Hz. However, because of the circuit resistor and some environment influence, the frequency will usually be lower than 494,948 Hz.

$$F = \frac{1}{2\pi\sqrt{LC}} = \frac{1}{2\pi\sqrt{220p * 470u}} = 494{,}948\ Hz$$

1.3 Demo Result

No ferromagnetic material closing	Ferromagnetic material closing
	
	

2 Code Description

2.1 Define and global variable

Define voltage threshold and claim global variable for counting ADC times.

```
/*---------------------------------------------------*/
/* Define                                            */
/*---------------------------------------------------*/
#define Threshold 0x850

/*---------------------------------------------------*/
/* Global variables                                  */
/*---------------------------------------------------*/
uint32_t g_u32ADCCount = 0;
```

2.2 ADC Interrupt

Every times ADC converts complete, entering ADC interrupt function for counting the times of over the threshold.

```
void ADC_IRQHandler(void)
{
    uint32_t u32Flag;
    // Get ADC comparator interrupt flag
    u32Flag = ADC_GET_INT_FLAG(ADC, ADC_CMP0_INT);
    // Add the count for calculate that how long the signal is bigger than threshold
    if(u32Flag & ADC_CMP0_INT)
        g_u32ADCCount++;
    ADC_CLR_INT_FLAG(ADC, u32Flag);
}
```

2.3 Main Function

Set the PC6 in output mode for generating bias voltage at resistor. Set the PA0 in open drain mode for keeping I/O on floating state. ADC will convert the voltage of PA0 for calculating the times of over the threshold to check if there is a metal object.

```
int32_t main (void)
{
    …
    GPIO_SetMode(PC,BIT6,GPIO_PMD_OUTPUT);
    PC6 = 1;//Output high level to generate bias voltage at resistor
    GPIO_SetMode(PA,BIT0,GPIO_PMD_OPEN_DRAIN);
```

```
    PA0 = 1;//Initial IO as floating state when PA0 is open drain mode


    // Enable channel 0
    ADC_Open(ADC, ADC_INPUT_MODE_SINGLE_END, ADC_OPERATION_MODE_CONTINUOUS,
ADC_CH_0_MASK);
    // Set reference voltage to AVDD
    ADC_SET_REF_VOLTAGE(ADC, ADC_REFSEL_POWER);


    while(1)
    {
        //Open ADC
        ADC_POWER_ON(ADC);
        //Start ADC Convertion
        ADC_START_CONV(ADC);


        //Open ADC comparator and set the threshold
        ADC_ENABLE_CMP0(ADC, 0, ADC_CMP_GREATER_OR_EQUAL_TO, Threshold, 1);
        //Enable ADC interrupt
        ADC_EnableInt(ADC, ADC_CMP0_INT);
        NVIC_EnableIRQ(ADC_IRQn);


        PA0 = 0; //Set PA.0 as Output low
        __NOP(); //Delay for capacitor charge
        __NOP();
        PA0 = 1; //Set PA.0 as floating state


        CLK_SysTickDelay(100); // Delay for LC sensor detection


        ADC_STOP_CONV(ADC);


        //Print the counter that bigger than threshold
        printf("g_u32ADCCount=%d\n",g_u32ADCCount);
        g_u32ADCCount = 0;
    }
}
```

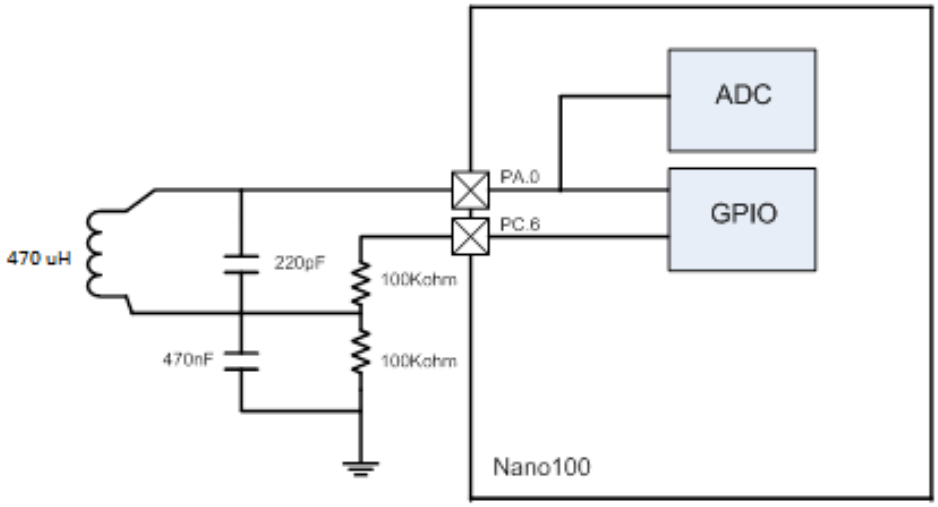
3 Software and Hardware Environment

Software Environment

- BSP version
  - Nano100B_Series_BSP_CMSIS_v3.02.002
- IDE version
  - Keil uVersion 5.26

Hardware Environment

- Circuit components
  - NuTiny-Nano100K
  - Capacitor : 470 nF *1, 220 pF *1
  - Inductance : 470 uH *1 (Inductance coil)
  - Resistor : 100 kOhm *2
- Diagram



4 Directory Information

- EC_Nano100_LC_Sensor_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. This project supports Keil uVersion 5.26 and higher version
2. According to the chapter 4 Directory Information, enter the ExampleCode/SampleCode/Keil folder and double click the LC_Sensor.uvproj
3. Enter the Keil compile mode window
   a. Build the project
   b. Download
   c. Start / Stop the debug mode
4. Enter debug mode
   a. Run

6 Revision History

Date	Revision	Description
Jun. 24, 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.

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.