Using ML56 Touch Slider to Control ARGB LED

Example Code Introduction for 8-bit NuMicro® Family

Information

Application	This sample code uses touch slider to control ARGB LED effect and uses SPI to simulate ARGB LED timing sequence on ML56 series.
BSP Version	ML51 Series BSP V2.00.001
Hardware	ML56 Demo Board Ver1.0

1 Overview

This sample code uses touch slider to control ARGB to do LED dimming and uses SPI to simulate ARGB LED timing sequence to achieve intuitive brightness control.

The ML56 demo board is a sensing board with a slider and a wheel. All of the sensors on this board are capacitive touch key sensing. The 6-element slider is connected to the following I/O: TK2(P29), TK3(P28), TK4(P27), TK5(P26), TK6(P25) and TK7(P24). When the finger touches the copper pad on the PCB, the current position can be reported by detecting the measurement of the capacitance changes. Slider supports 10, 20, 50, 100 resolutions and the calibration parameters are shown in Figure 1-1. In the case, slider with 100 points of resolution should report position in the range of 0 -100. ARGB LED brightness and the slider position can be 0 to 100 corresponding 0 to 100% brightness.

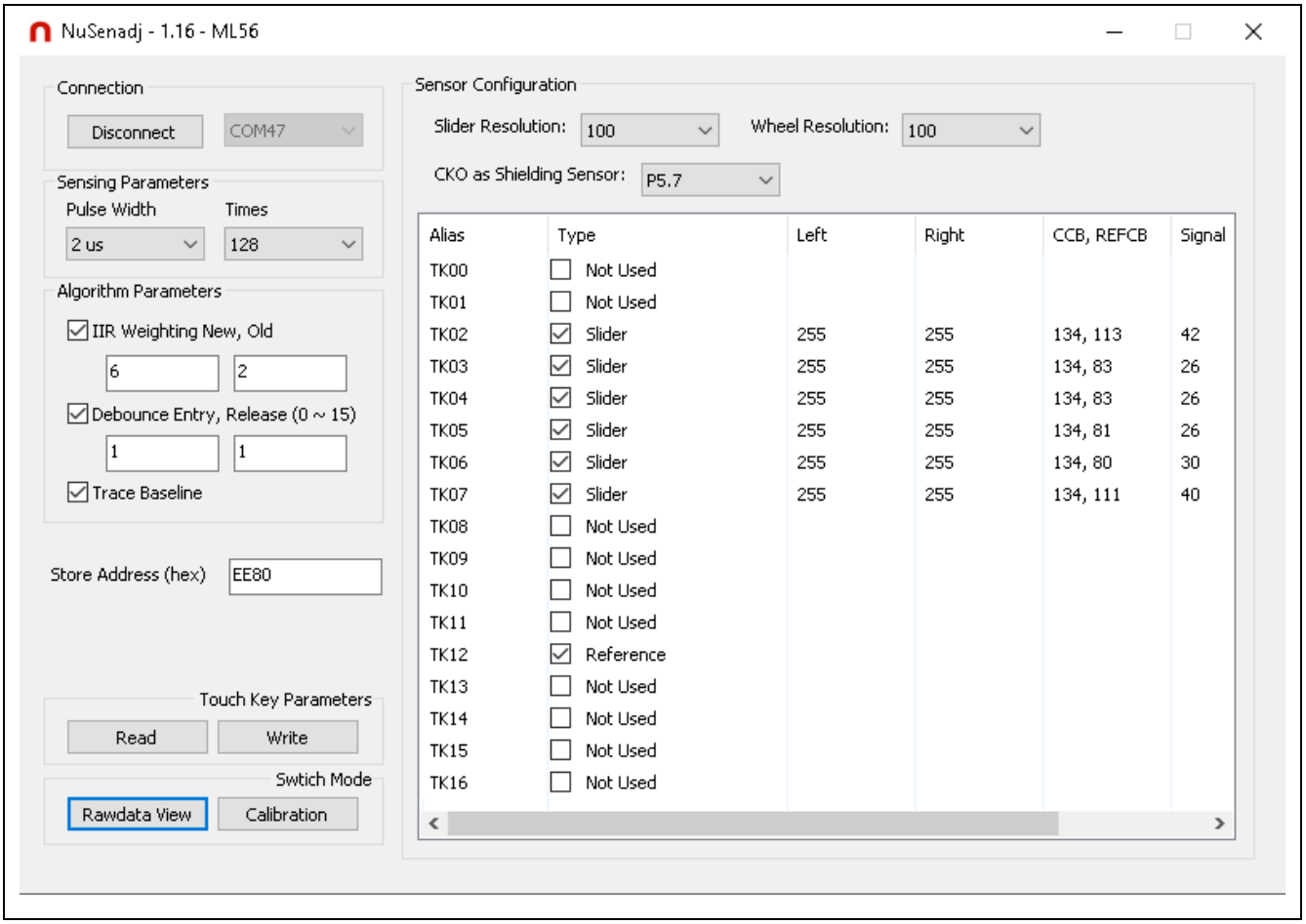

Figure 1-1 Calibration Parameters of ML56 Demo Board

1.1 Principle

1.1.1 ARGB LED Data Transfer Time (TH+TL=1.25 us ± 600 ns)

ARGB LED timing sequences are shown in Figure 1-2 and Figure 1-4. Logic 0 is defined as a high pulse of 0.4 us ± 150 ns followed by a low pulse of 0.85 us ± 150 ns. Logic 1 is defined as a high pulse of 0.8 us ± 150 ns followed by a low pulse of 0.45 us ± 150 ns. Reset is defined as a low pulse above 50 us.

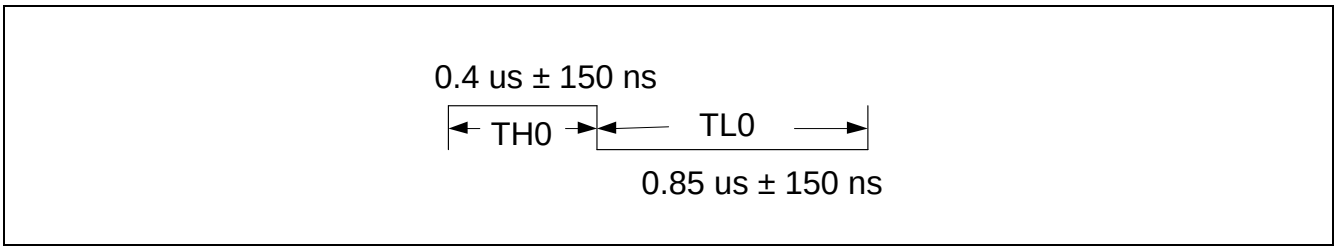

Figure 1-2 Logic 0

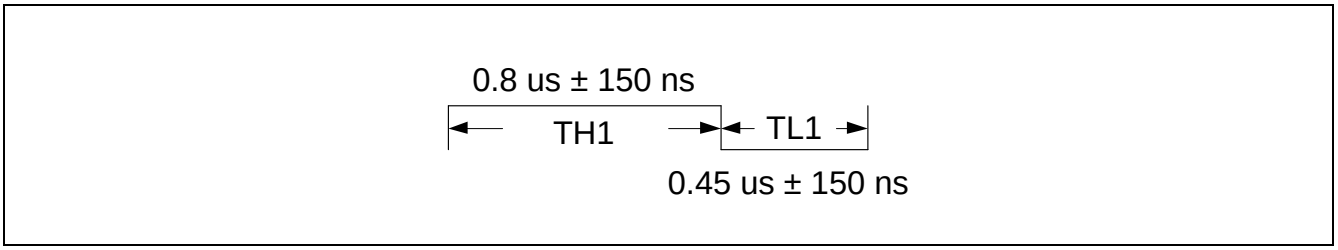

Figure 1-3 Logic 1

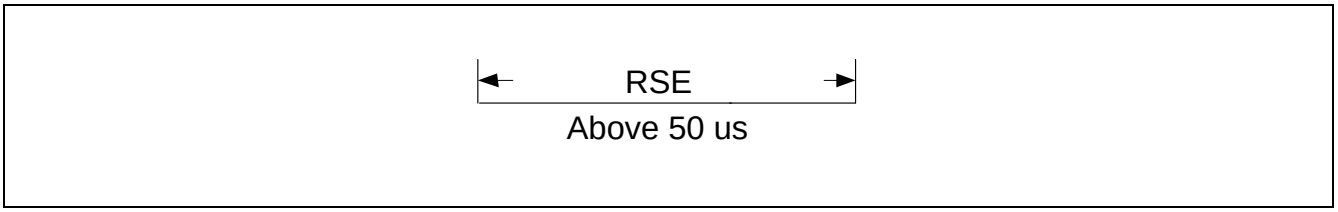

Figure 1-4 Reset

1.1.2 SPI Clock Frequency

When SPI clock divider is 8 and HIRC clock is 24 MHz, the theoretical value for the SPI clock is 3 MHz. It means that one SPI bit lasts 0.333 us. In other words, logic 0 is mapped to SPI bit pattern of ‘1000’ and logic 1 is mapped to SPI bit pattern of ‘1110’. The 3-byte Green-Red-Blue value is converted to 12-byte SPI sequence.

1.2 How to Use NuSenadj

Please follow the steps below.

1. Download the *calibration.bin* file using ICP programming tool as shown in Figure 1-5. The binary file is stored in TK-Utility folder.

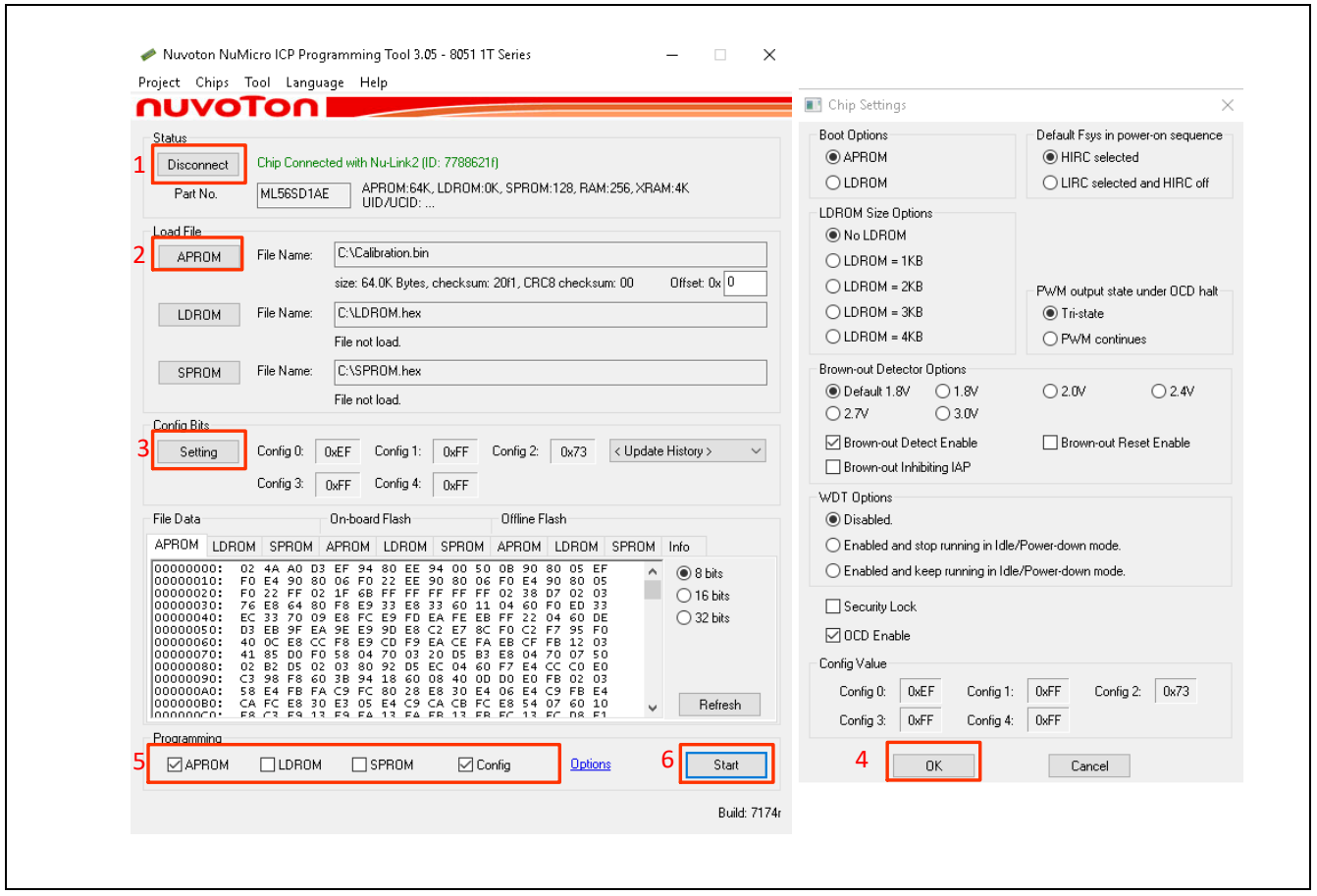

Figure 1-5 Programming Calibration.bin File

2. To set calibration parameters and start calibration process, use *NuSenadj.exe* as shown in Figure 1-6 and Figure 1-7.

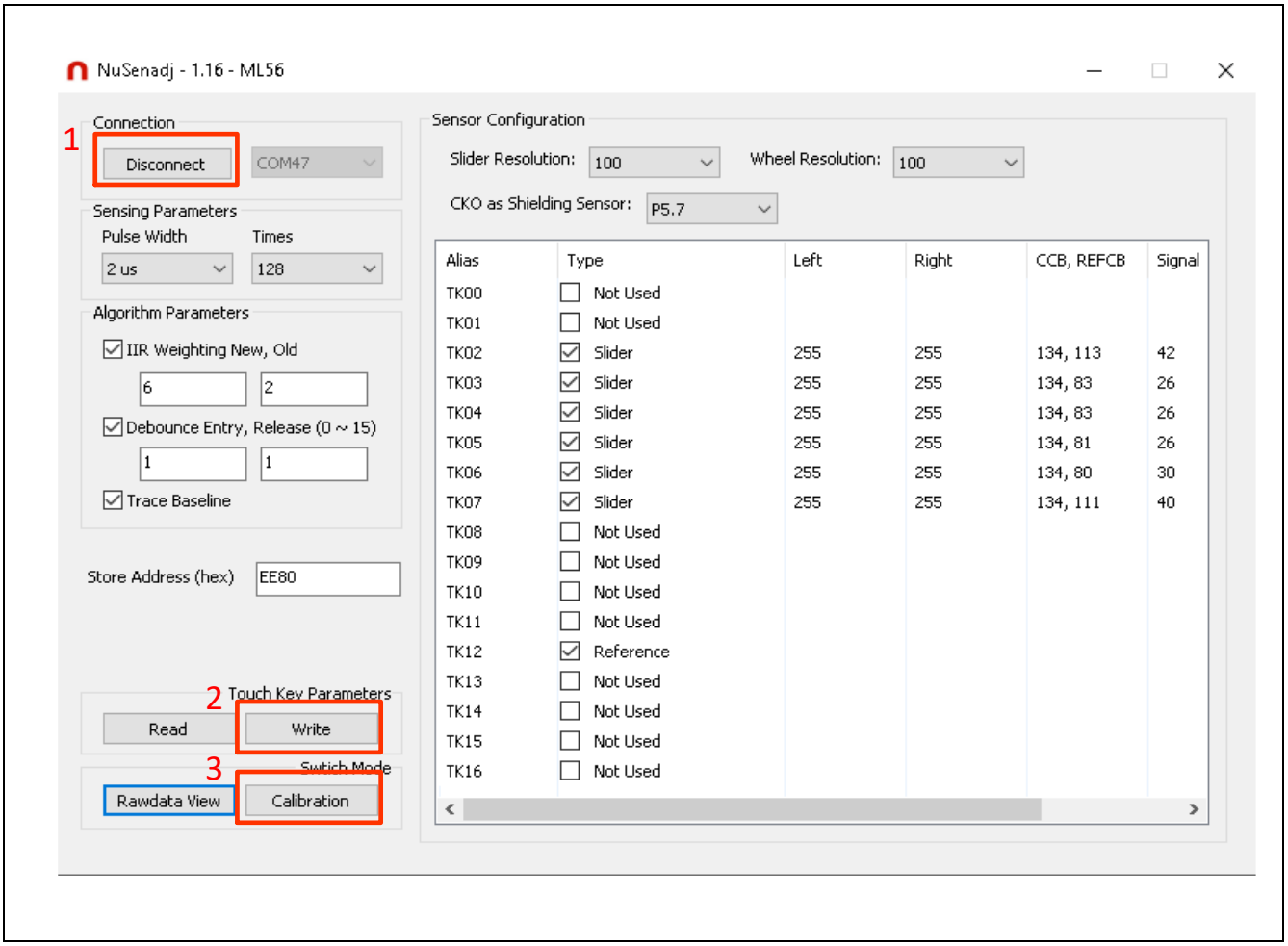

Figure 1-6 Set Calibration Parameters

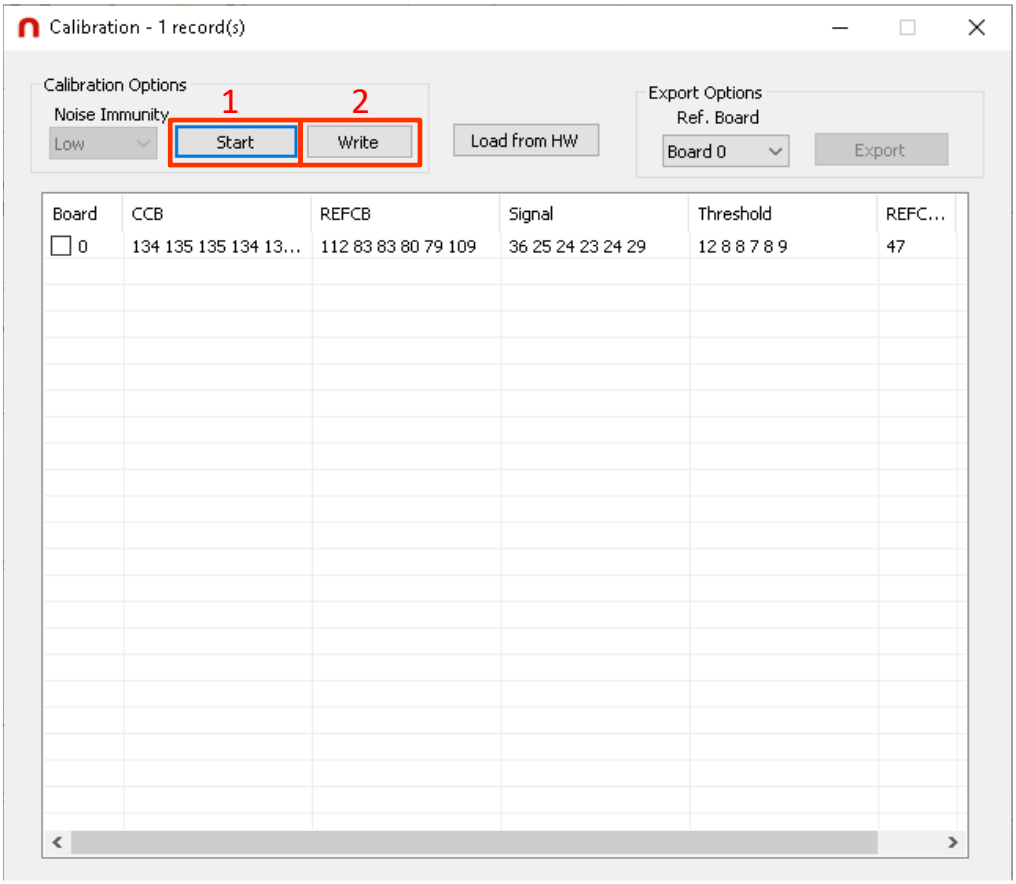

Figure 1-7 Start Calibration Process

3. Download the ML56_Slider_ARGB project using Keil.

1.3 Demo Result

The demo result is shown in Figure 1-8.

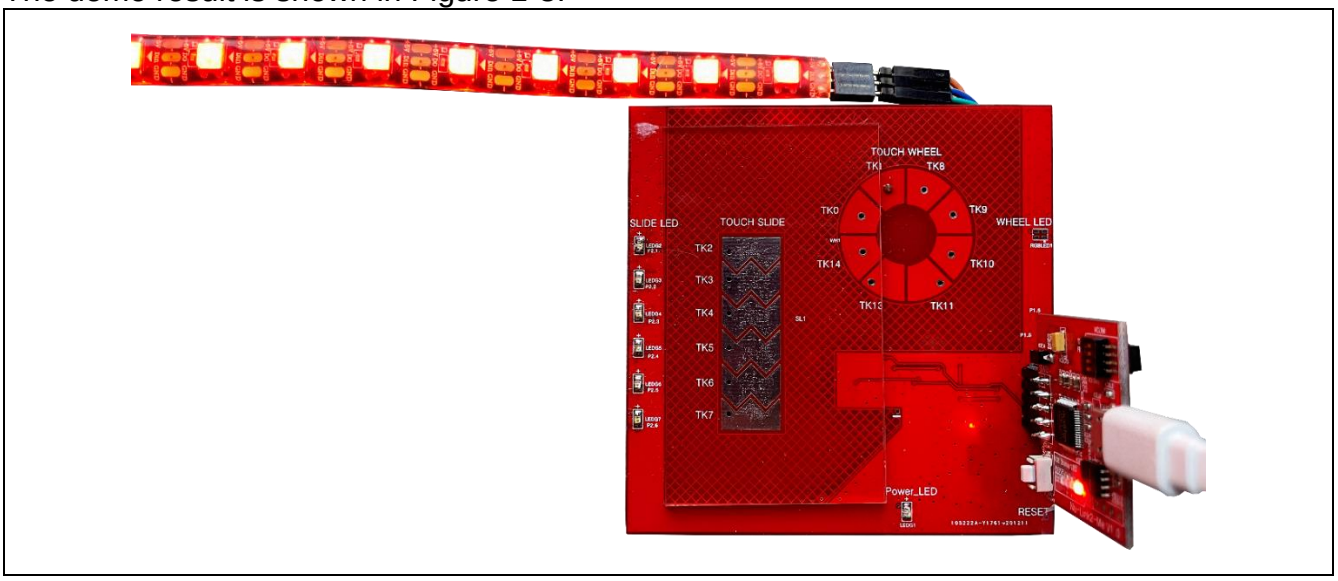

Figure 1-8 Demo Result

2 Code Description

The instructions are listed below:

1. Initialize the ML56 in main.c, including FMC, touch key, SPI, and Timer 2.
   - FMC is used to load touch sensor setting and calibration data from Flash.
   - Touch key is used for slider function.
   - SPI is used to simulate ARGB timing.
   - Timer2 is used to set the touch sensor scan rate at 20 ms.
2. In while-loop, call TK_RawDataView() function to get slider current position, which is then passed into Slider_TOPDOWN() function to calculate brightness.
3. Use SPI_Send() function to control ARGB LED.

```
int32_t main(void)
{
    uint32_t u32ChanelMsk;
    int8_t i8Ret = 0;
    uint8_t rate;

    /* Initialize FMC to Load TK setting and calibration data from flash */
    FMC_Open();
    i8Ret = TK_LoadPara(&u32ChanelMsk);

    /* Fail to load TK setting and calibration data from flash */
    if(i8Ret < 0)
    {
        while(1);
    }

    /* Init TK Controller */
    TK_Init();

    /* Initialize Multiple Function Pins for TK */
    SetTkMultiFun(u32ChanelMsk);

    /* Init systick 20ms/tick */
    Init_SysTick();

    /* Install Tick Event Handler To Drive Key Scan */
    TickSetTickEvent(1, TickCallback_KeyScan);
    TickSetTickEvent(5, TickCallback_SliderScan);

    /* Init SPI Controller */
    SPI_Init();

    /* Enable all interrupt */
    set_IE_EA;

    /* Init LED to all disable */
    Disable_TOPDOWN();
    SPI_Send();
    do
    {
        /* Report slider position */
```

```
        TK_RawDataView();

         /* Control LED */
        if (spi_finish_flag == 0 )
        {
             /* Calculate LED brightness */
            Slider_TOPDOWN(i8SliderPercentage);

             /* Using SPI to control LED strip */
            SPI_Send();
        }

    }
    while (1);
    return 0;
}
```

When the TK_RawDataView() function is called, it will get slider position. The slider scan rate at 20 ms by default.

```
void TK_RawDataView(void)
{
    /* The buffer size is equal to the ML56 TK channels*/
    xdata int8_t ai8Signal[TKLIB_TOL_NUM_KEY];
    int8_t ai8TmpSignal[TKLIB_TOL_NUM_KEY];

    if (u8EventKeyScan == 1)
    {
        uint8_t i;
        u8EventKeyScan = 0;

         /* Scan all enable key, slider and wheel channels */
        i8Ret = TK_ScanKey(&ai8Signal[0]);

#if defined(OPT_SLIDER)
        {
            /** To save buffer size, re-used the ai8Signal[] buffer
              * Remember that the buffer will be destroied
              */
            uint16_t u16ChnMsk; /* ML56 is only 15 TK channels */
            uint8_t u8Count = 0, i;
            static uint8_t updatecount = 0;

            updatecount = updatecount+1;
            if(updatecount < 5)
                return;
            updatecount = 0;

             /* Get slider channels */
            u16ChnMsk = TK_GetEnabledChannelMask(TK_SLIDER);

             /* Check slider be pressed */
            if(TK_CheckSliderWheelPressed(TK_SLIDER) == 1)
            {
                for (i = 0; i < TKLIB_TOL_NUM_KEY ; i++)
                {
                    if (u16ChnMsk & (1ul << i))
                    {
                        ai8TmpSignal[u8Count] = ai8Signal[i];
                        u8Count = u8Count+1;
                    }
                }
                i8SliderPercentage = TK_SliderPercentage(ai8TmpSignal, u8Count);
            }
```

```
        }
#endif
    }
}
```

The function of Slider_TOPDOWN() that takes a slider_percentage as parameter. The ARGB LED brightness and slider_percentage can be 0 to 100 corresponding 0 to 100% brightness. When the function is called, the ARGB LED brightness will be calculated periodically.

```
void Slider_TOPDOWN(unsigned int slider_percentage)
{
    unsigned char TempR, TempG, TempB;
    unsigned long TempCal;

    /* Calculate brightness */
    TempCal = *(TestBright + (HDIV_Mod(HDIV_Div(slider_percentage, 2), TestArraySize)));
    TempCal = HDIV_Div((TempCal * TOPDOWNBrightness), 200);

    /* Calculate color and set transfer data */
    TempR = HDIV_Div((TOPDOWNRGBColor[0] * TempCal), 200);
    TempG = HDIV_Div((TOPDOWNRGBColor[1] * TempCal), 200);
    TempB = HDIV_Div((TOPDOWNRGBColor[2] * TempCal), 200);
    Set_TOPDOWN_RGB(TempR, TempG, TempB);
}
```

3 Software and Hardware Requirements

3.1 Software Requirements

- BSP version
  - ML51 Series BSP v2.00.001
- IDE version
  - Keil uVersion 5.26

3.2 Hardware Requirements

- Circuit components
  - ML56 Demo Board Ver1.0
  - ARGB LED

- Pin Connect

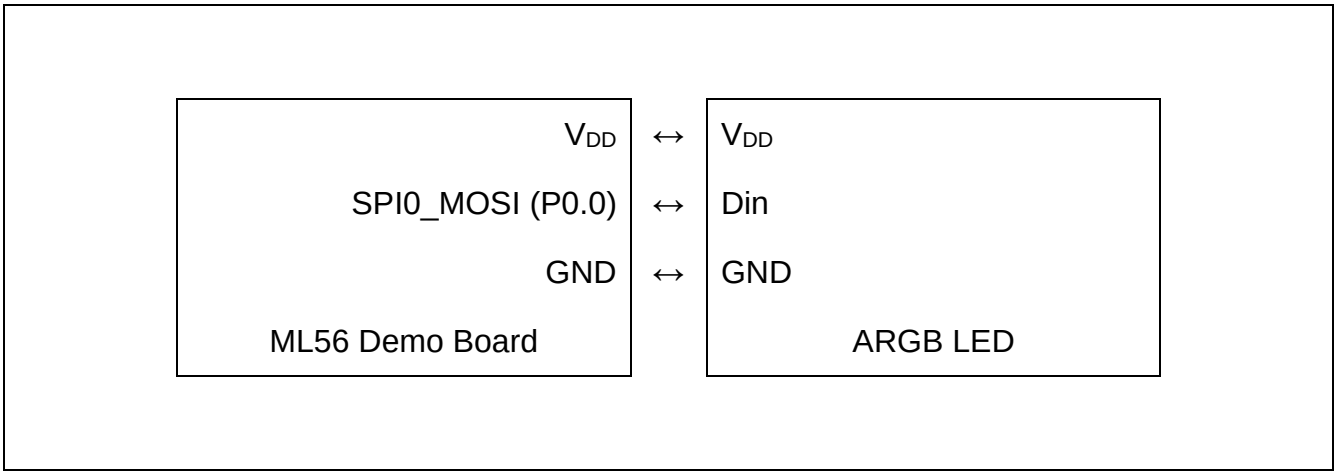

Figure 3-1 Pin Connect

4 Directory Information

The directory structure is shown in Figure 4-1.

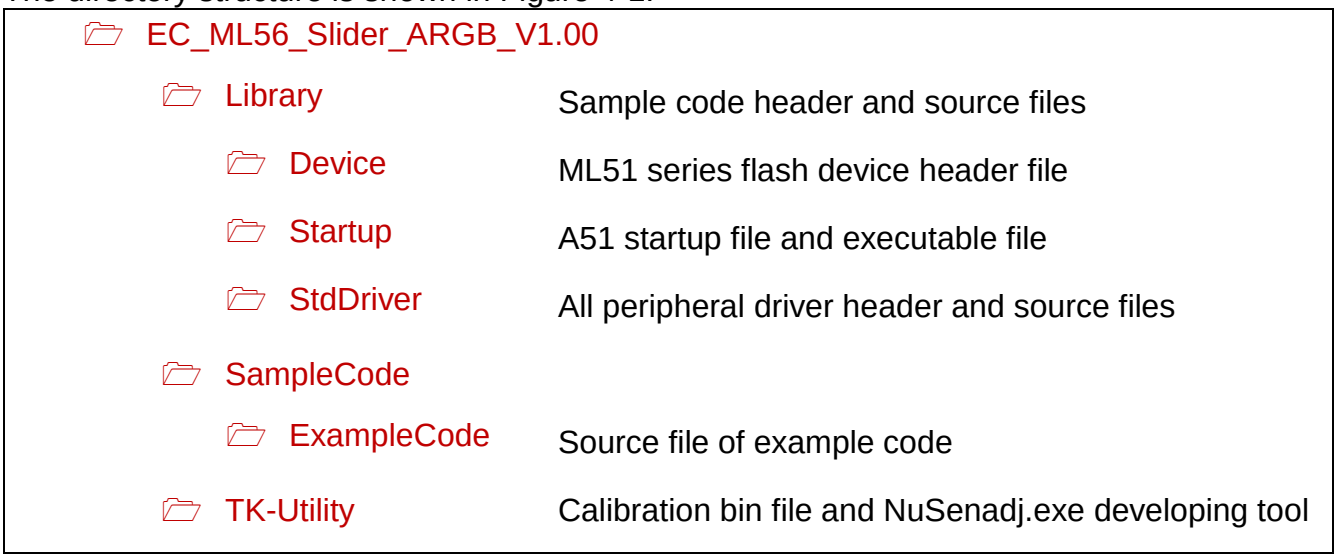

Figure 4-1 Directory Structure

5 Example Code Execution

1. Browse the sample code folder as described in the Directory Information section and double-click *ML56_Slider_ARGB.uvproj*.
2. Enter Keil compile mode.
   - Build
   - Download
   - Start/Stop debug session
3. Enter debug mode.
   - Run

6 Revision History

Date	Revision	Description
2021.07.29	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.