Using MS51 SPI to Realize Rainbow Wave ARGB LED Effect

Example Code Introduction for 8-bit NuMicro® Family

Information

Application	This sample code uses SPI to simulate ARGB LED timing sequence and realize Rainbow Wave ARGB LED effect on MS51 series.
BSP Version	MS51 Series BSP V1.00.003
Hardware	NuMaker-MS51PC Ver1.0

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 and microprocessor 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 Overview

This sample code uses SPI to simulate ARGB LED timing sequence and do LED Rainbow Wave effect. This code uses one timer to schedule ARGB LED to do ARGB LED toning and one set of SPI to simulate ARGB LED strip timing sequence.

1.1 Principle

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

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

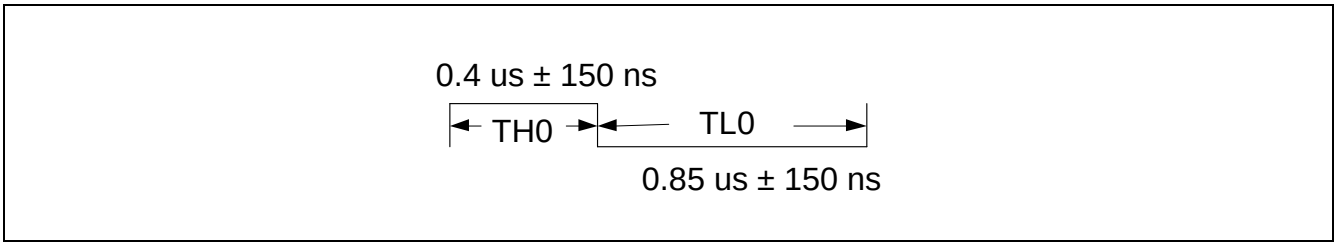

Figure 1-1 Logic 0

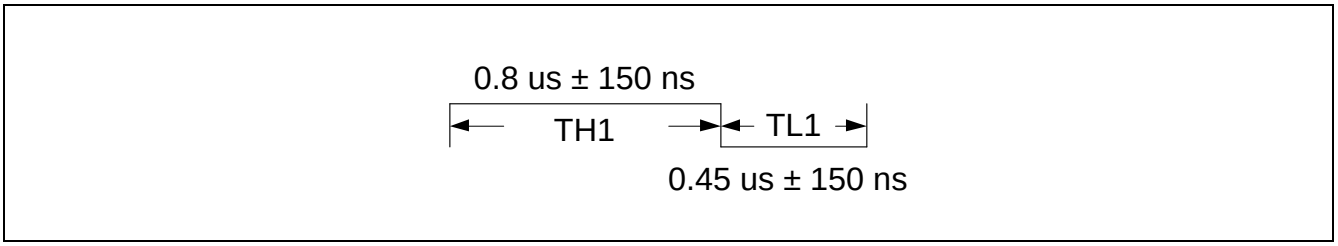

Figure 1-2 Logic 1

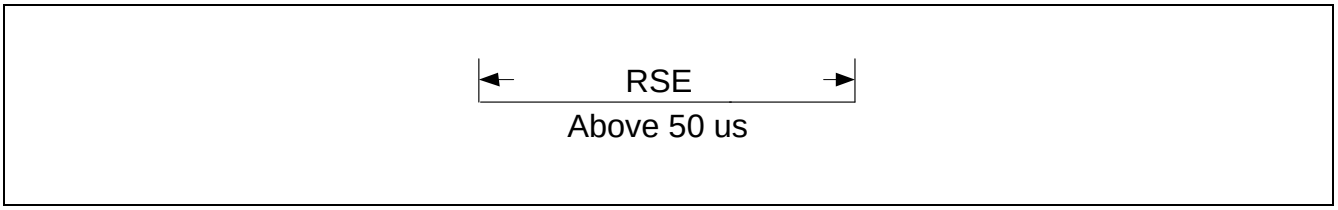

Figure 1-3 Reset

1.1.2 SPI Clock Frequency

When SPI clock divider is 8 and HIRC clock is 24 MHz, the theoretical for the SPI clock is 3 MHz. It means that one SPI bit lasts 0.333us. 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.

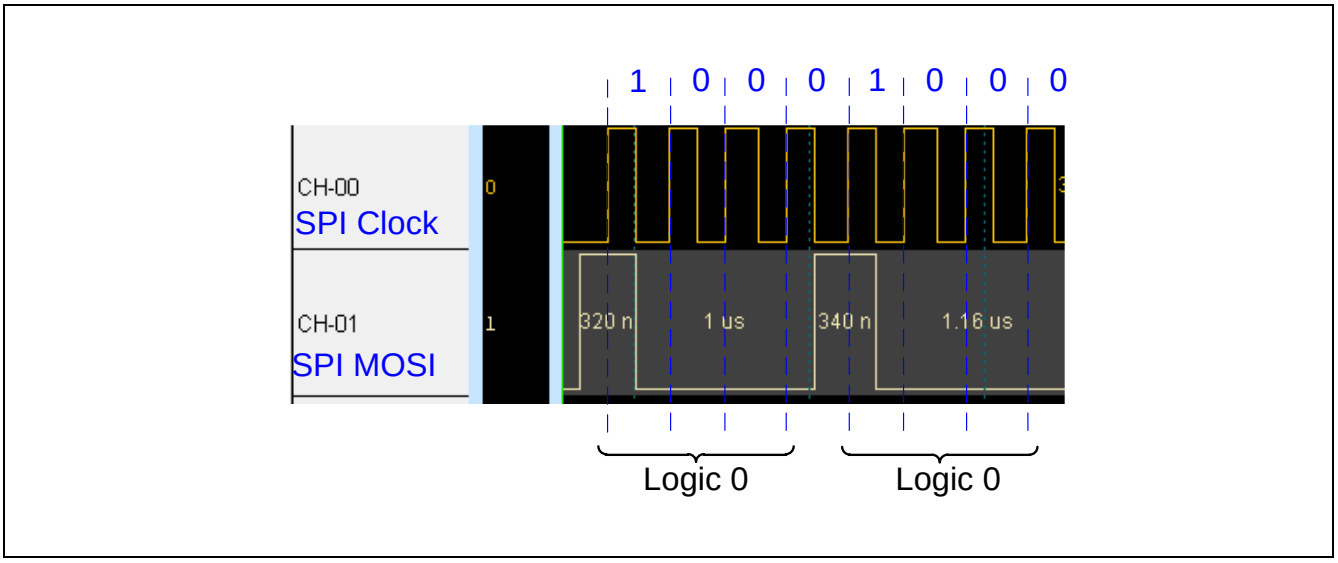

Figure 1-4 Logic 0 Is Mapped to SPI Bit Pattern

1.2 Demo Result

The demo result is shown below.

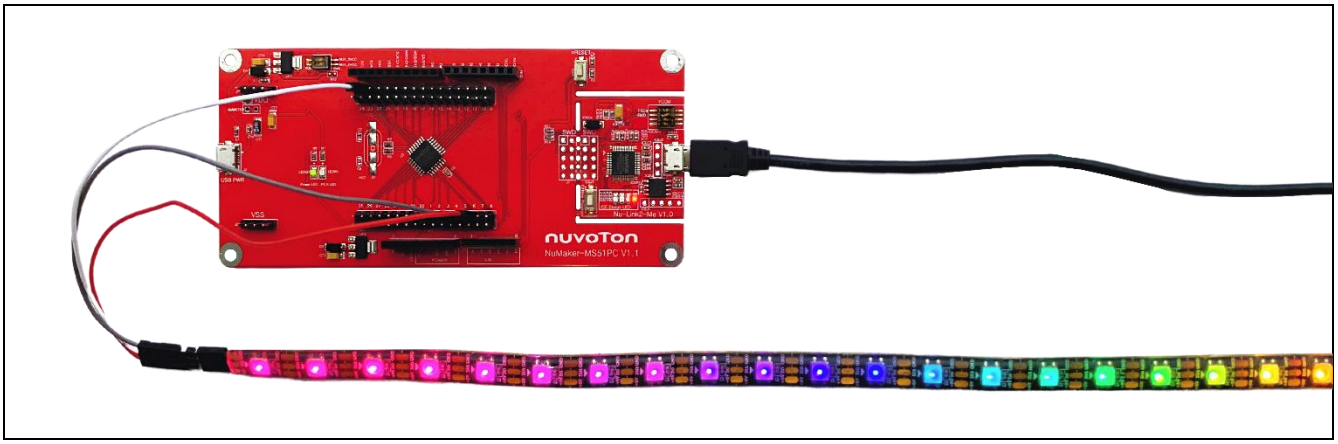

Figure 1-5 Demo Result

2 Code Description

The instructions are listed below:

1. Initialize the MS51 in main.c, including HIRC, SPI, and Timer 3.
2. In while-loop, call RainbowWave_TOPDOWN() function to calculate color.
3. Use SPI_Send() function to control ARGB LED.

```
void main (void)
{
    MODIFY_HIRC(HIRC_24);

    /* Set LED output */
    P00 = 1;
    P00_INPUT_MODE;
    ENABLE_P00_PULLDOWN;

    SPI_Init();

    /* Enable Timer3 to add time counter */
    Timer3_1ms_initial();


    /* Init LED to all disable */
    Disable_TOPDOWN();
    SPI_Send();

    while(1)
    {

    /* Control LED */
        {
            RainbowWave_TOPDOWN();
            SPI_Send();
        }
    }
}
```

When RainbowWave_TOPDOWN() function is called, ARGB LED color is calculated periodically.

```
void RainbowWave_TOPDOWN(void)
{
    unsigned char i, j;
    /* Velocity = LEDs / Second */
    unsigned char Velocity[3] = {5, 9, 13};
    unsigned long deltaT, TotalDuration, CurrentTime, NextTime;
    unsigned char ColorCursor1, ColorCursor2;

    /* Duration for one time effect */
    TotalDuration = HDIV_Div(30 * 1000, Velocity[TOPDOWNSpeed]);
    /* Next LED time */
    NextTime = HDIV_Div(TotalDuration, 30);
    /* Current Time */
    CurrentTime = HDIV_Mod(TOPDOWNTimeCounter, TotalDuration);
    /* Unit time for each color */
    deltaT = HDIV_Div(TotalDuration, 7);
```

```
    /* Reset CountingTime */
    if(TOPDOWNTimeCounter >= TotalDuration)
    {
        TOPDOWNTimeCounter -= TotalDuration;
    }

    /* Mix Color */
    for(j = 0; j < TOP_DOWN_LED_Num; j++, CurrentTime += NextTime)
    {
        if(CurrentTime >= TotalDuration)
            CurrentTime -= TotalDuration;

        if(CurrentTime >= (deltaT * 7))
        {
            /* Color Cursor */
            ColorCursor1 = 0;
            ColorCursor2 = 1;

            for(i = 0; i < 3; i++)
            {
                TOPDOWNRGBArray[(TOP_DOWN_LED_Num-j-1)][i]=RainbowColor[ColorCursor1][i];
            }
        }
        else
        {
            /* Color Cursor */
            ColorCursor1 = HDIV_Mod(HDIV_Div(CurrentTime, deltaT), 7);
            ColorCursor2 = HDIV_Mod((ColorCursor1 + 1), 7);

            for(i = 0; i < 3; i++)
            {
                TOPDOWNRGBArray[(TOP_DOWN_LED_Num-j-1)][i] =
HDIV_Div(((RainbowColor[ColorCursor1][i] * ((ColorCursor1 + 1) * deltaT - CurrentTime)) +\

(RainbowColor[ColorCursor2][i] * (CurrentTime - ColorCursor1 * deltaT))), deltaT);
            }
        }
    }
}
```

3 Color Calculation Formula

Calculate the RGB mixing ratio of the front and back colors based on the ratio of the area.

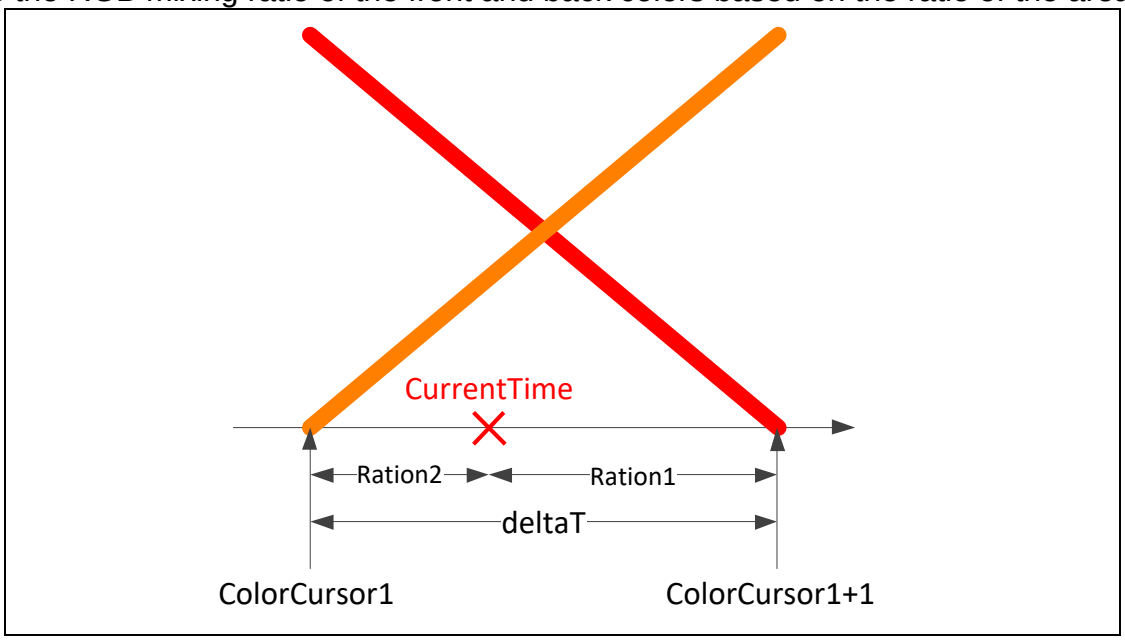

Figure 3-1 Calculate the RGB Mixing Ratio Motion Diagram

Ration1 = (ColorCursor1 + 1) * deltaT – CurrentTime
Ration2 = CurrentTime - ColorCursor1 * deltaT

Color Calculation Formula:

$$Color = \frac{Color1 \times Ration1}{deltaT} + \frac{Color2 \times Ration2}{deltaT} =$$

$$\frac{Color1 \times \left((ColorCursor1+1) \times deltaT - CurrentTime\right) + Color2 \times (CurrentTime - Cursor1 \times deltaT)}{deltaT}$$

4 Software and Hardware Requirements

4.1 Software Requirements

- BSP version
  - MS51 Series BSP v1.00.003
- IDE version
  - Keil uVersion 5.26

4.2 Hardware Requirements

- Circuit components
  - NuMaker-MS51PC Ver1.0
  - ARGB LED

- Pin Connect

	VCC	↔	VCC	
	SPI0_MOSI (P0.0)	↔	Din	
	GND	↔	GND	
	MS51		ARGB LED	

Figure 4-1 Connect diagram

Note: The operating voltage of ARGB LED is 5V, so the power supply voltage of NuMaker-MS51PC for the ARGB LED must be changed to 5V. Please switch ICEJPR1(MCUVCC) to make Nu-Link2-Me output 5V voltage.

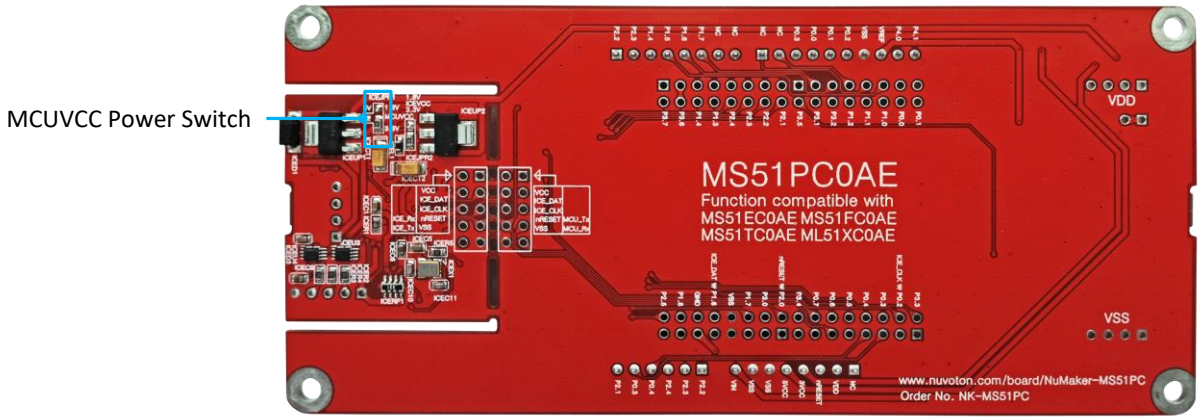

Figure 4-2 MCUVCC Power Switch to 5V

5 Directory Information

The directory structure is shown below.

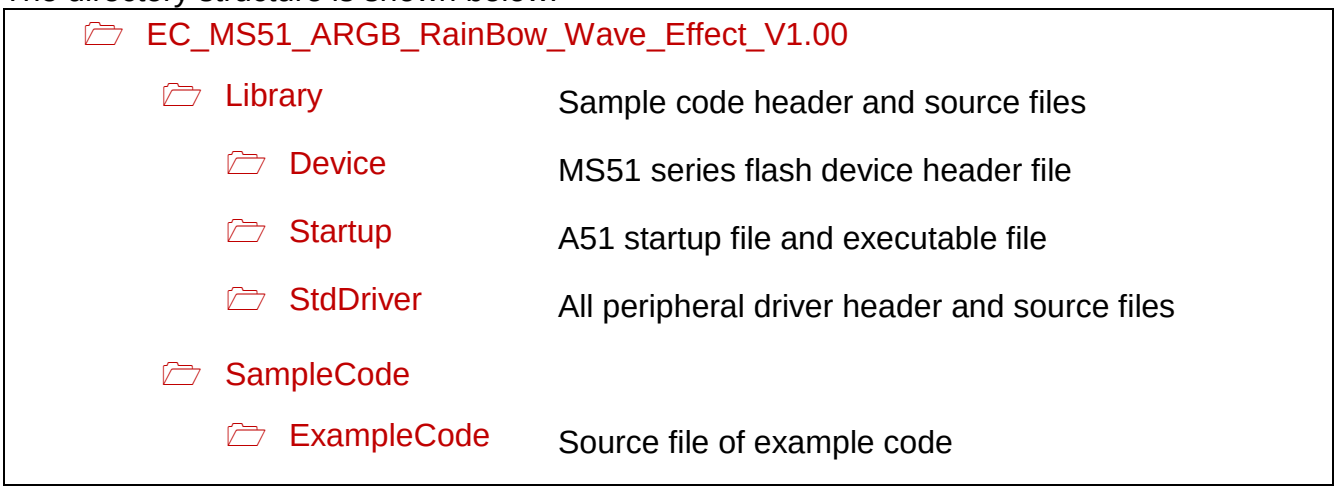

Figure 5-1 Directory Structure

6 Example Code Execution

1. Browse the sample code folder as described in the Directory Information section and double-click *MS51_ARGB_RainBow_Wave_Effect.uvproj*.
2. Enter Keil compile mode.
   - Build
   - Download
   - Start/Stop debug session
3. Enter debug mode.
   - Run

7 Revision History

Date	Revision	Description
2021.08.20	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.