Application Note

Laser Speckle Reduction Solution in Projector Based on MSPM0 MCU

Helic Chi, Zoey Wei, and Rocky Chen

ABSTRACT

As a general-purpose MCU product, MSPM0 has the feature of complete peripherals and high cost performance, making the Laser Speckle Reduction more flexible and low-cost. This application note is a guide for the implementation of Laser Speckle Reduction based on MSPM0, including several methods to generate the required PWM waveform and test result. In addition, this document provides a method to calibrate the event and interrupt delay between peripherals.

Table of Contents

Trademarks

1 Introduction

In projectors, there are several light source technologies. Among them, triple color laser is a projector light source technology with high brightness, wide color gamut and long life. Lasers have inherently high coherence. When lasers are irradiated onto optically rough surfaces, the coherent light interferes and produces a granular speckle pattern with light and dark transitions. Laser speckle affects the image contrast and resolution. Therefore, in three color projector application, laser speckle reduction technology is needed.

1.1 Laser Speckle Reduction

Laser Speckle Reduction (LSR), is a technology to reduce the laser speckle. In three color projector, there are several methods to reduce the laser speckle,

- Process laser light at the source.
- In the optical path, superimpose the random speckles to reduce the intensity of the speckles.

The common speckle superposition method is vibration. In the optical path, there are two general places to apply vibration,

- Optical machine vibration.
- Screen vibration.

Home and portable projectors are hard to add vibration to the screen, which increases the installation cost of the projector. Therefore, adding vibration to the projector optical machine is a low-cost method to reduce speckle.

Vibration is generally generated using coils or motors. This application note introduces the methods to generate the coils control PWM waveform using MSPM0.

1.2 MSPM0 Requirements

In this application note, use DRV8847 to control two coils, and each coil controls the optical machine to vibrate on the X-axis and Y-axis, respectively. By controlling the vibration intensity and phase of each coil, DRV8847 can control the vibration direction and distance of the optical machine in the X-Y plane.

DRV8847 is a dual H-bridge motor driver with current regulation and independent half-bridge con. Figure 1-1 is two groups of PWM waveform signals that using to drive two coils. For the procedure that DRV8847 processes these PWM signals, refer to DRV8847 block diagram and DRV8847 Dual H-Bridge Motor Driver data sheet.

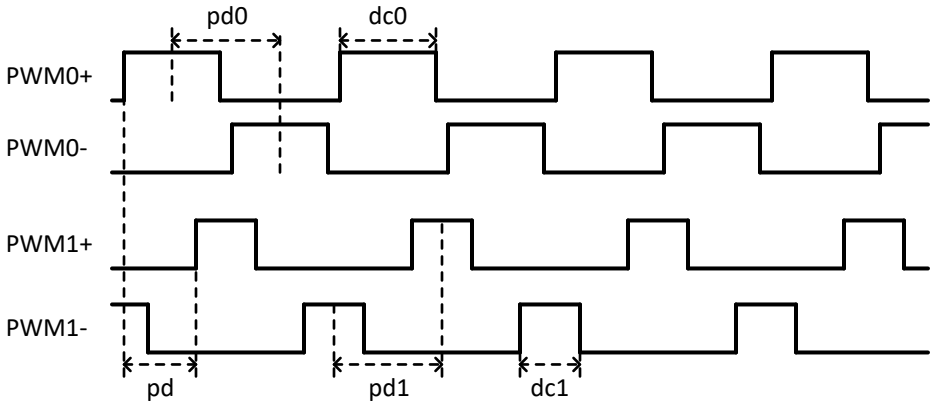

Figure 1-1. PWM Waveform

- *PWM0+* and *PWM0-* are used to drive coil0.
- *PWM1+* and *PWM1-* are used to drive coil1.
- *pd0* is the phase difference between *PWM0+* and *PWM0-*, fixed value 180°.
- *pd1* is the phase difference between *PWM1+* and *PWM1-*, fixed value 180°.
- *pd* is the phase difference between *PWM0* and *PWM1*, adjustable from -90° to 90°.
- *dc0* is the duty cycle of *PWM0+* and *PWM0-*, adjustable from 0% to 100%.
- *dc1* is the duty cycle of *PWM1+* and *PWM1-*, adjustable from 0% to 100%.

Depending on the device selection, there are multiple ways to generate the above PWM waveform. The main peripherals used include Timer, Event and GPIO. Table 1-1 is the introduction and peripherals used of different methods. For detailed description, please refer to each method's section.

Table 1-1. LSR Control Methods Summary

Method	Peripherals	MSPM0 Device Support	Description
PWM	Timer * 4	MSPM0 L and G series	Hardware method.
PWM and GPIO	Timer *3 GPIO * 1	MSPM0 C series	Hardware and software mixed control.
Timer and GPIO	Timer * 1 GPIO * 4	MSPM0 C, L and G series	Software method, high CPU clock is better for precise control.

Note

Hardware implement requires an event channel to trigger between event publisher and event subscriber, the delay of event trigger various from peripherals' power domain and peripherals' type. This kind of event delay is fixed, a fixed offset can be considered to be added to timer's CC (capture and compare) value if high-precision control is required. This suggestion applies to both event between two timers and between GPIO and timer.

Note

MSPM0 can output a variety of PWM signals based on timer peripheral, by using hardware timer with PWM or software IO control. MSPM0's timer support 4 or 2 CCs (capture and compare channel) and each CC can be used to output a PWM waveform or trigger a fixed delay via event channel. Hardware event feature supports hardware phase control between PWMs and duty cycle control of PWM. Each timer has two event publishers and one event subscriber. An event publisher can be used to generate a trigger to event channel, and an event subscriber can be used to receive a trigger from an event channel. Detailed information can be found from MSPM0 device's data sheet and Technical reference manual's Timer section.

2 PWM

2.1 PWM Implementation

The PWM method is a fully hardware implementation, using a timer to output PWM and an event to support a hardware trigger between the timer controller and timer target.

In the MSPM0 G-series, the PWM method requires four timers, including one timer with four CCs, and three timers with two CCs. Use MSPM0G3507 for example. Figure 2-1 shows the internal hardware control chain of timers and events of this method.

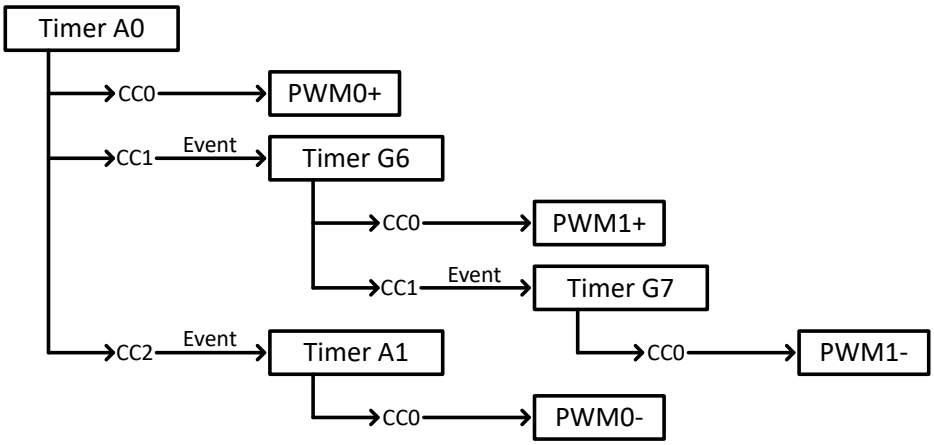

Figure 2-1. PWM Method Control Chain - G series

And, in MSPM0 L-series, because there is no timer with four CCs, four timers with two CCs can be used to output PWM waveform and four timers are triggered by event serially. Use MSPM0L1306 for example. Figure 2-2 shows the internal hardware control chain of timers and events of this method.

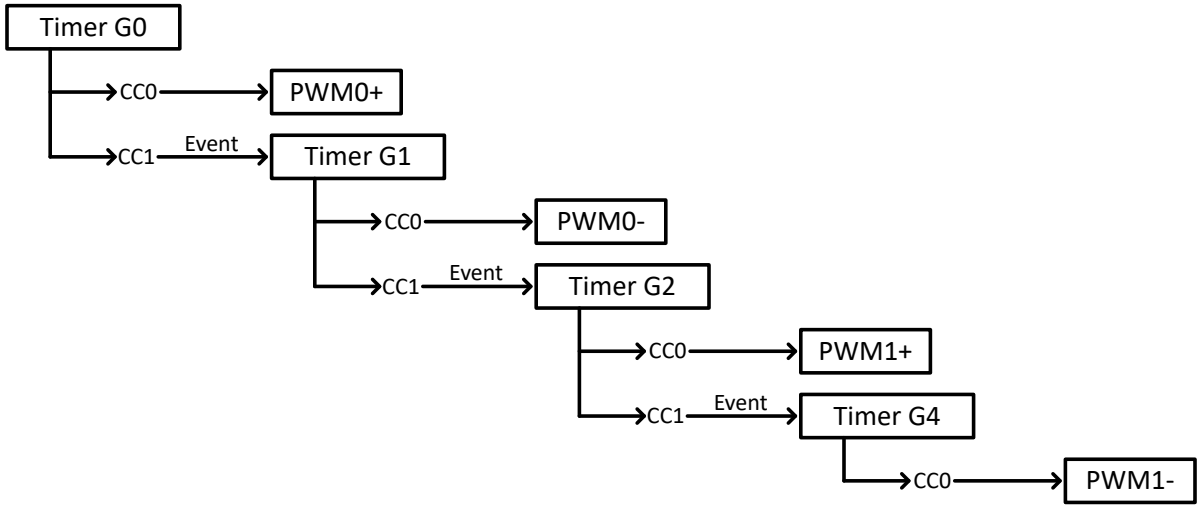

Figure 2-2. PWM Method Control Chain - L series

2.2 PWM Test Result

Figure 2-3 shows the test result of PWM method.

The phase difference can be controlled by adjusting the CC value of each timer:

- Timer A0 CC1 controls the *pd* value, which is the phase difference between *PWM0* and *PWM1*.
- Timer A0 CC2 controls the *pd0* value, which is fixed at 180°, phase difference between *PWM0+* and *PWM0-*.
- Timer G6 CC1 controls the *pd1* value, which is fixed at 180°, phase difference between *PWM1+* and *PWM1-*.

The duty cycle can be controlled by adjusting the CC value of each timer's CC0:

- Timer A0 CC0 and Timer A1 CC0 control the *dc0*, which is *PWM0+* and *PWM0-*'s duty cycle.
- Timer G6 CC0 and Timer G7 CC0 control the *dc1*, which is *PWM1+* and *PWM1-*'s duty cycle.

Also, by changing these timer load value, MSPM0 can output different period control PWM waveform.

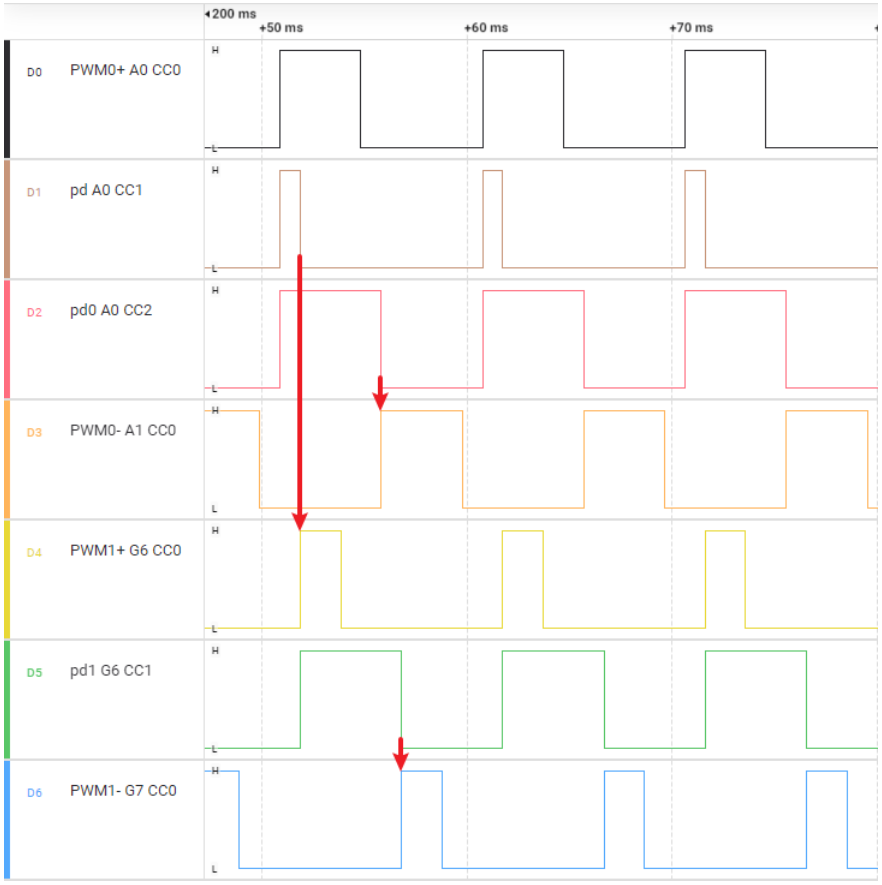

Figure 2-3. PWM Method Test Result

3 PWM and GPIO

3.1 PWM and GPIO Implementation

In MSPM0 C-series, such as MSPM0C1104, there are only three timers available, Timer A0, Timer G14, and Timer G8. MSPM0 needs another GPIO to output the last channel PWM. Figure 3-1 shows the internal hardware control chain of timers, events, and interrupt of this method.

Between timers, an event is still used as the hardware trigger method to control timers synchronously. A group of GPIO, such as GPIO A, supports one event to subscribe the channel, and one channel to support one GPIO action (set, clear or toggle). The MSPM0 needs additional timer interrupt to clear the GPIO while using an event to set GPIO.

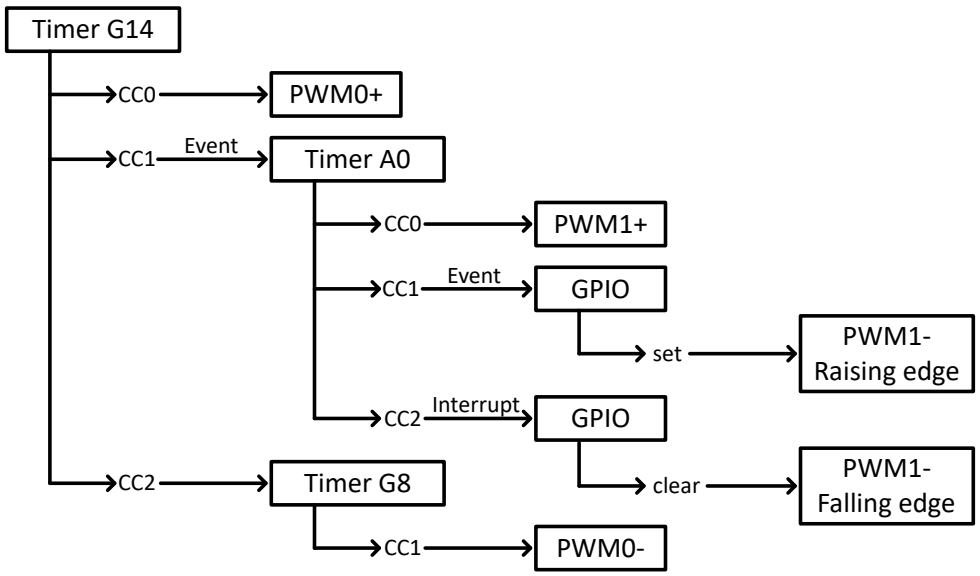

Figure 3-1. PWM and GPIO Method Control Chain

3.2 Interrupt Time Calibration

In the Figure 3-1, there is a interrupt control between Timer A0 CC2 and GPIO. Interrupt is a software control method and there is a large delay in using interrupt compared with event hardware control method.

Figure 3-2 shows the delay between Timer A0 CC2 PWM falling edge and GPIO falling edge. The red mask area is the delay time between PWM and GPIO. Code below shows the interrupt handling process, this delay is a fixed value if this timer interrupt is the only highest priority interrupt in the system. Timer A0 CC2's value need to be set to:

$$CC2 = CC2th - delay \quad (1)$$

where *CC2th* is the theoretical value of Timer A0 CC2, and *delay* is the red mask area shown in Figure 3-2. *delay* need to convert to timer CC value depending on the timer clock configuration.

Figure 3-2. Delay between PWM and GPIO

```
void PWM_1_P_INST_IRQHandler(void)
{
    switch(DL_Timer_getPendingInterrupt(PWM_1_P_INST)){
        case DL_TIMER_IIDX_CC2_UP:
            DL_GPIO_clearPins(PWM_1_N_PORT, PWM_1_N_PIN_0_PIN);
            break;
        default:
            break;
    }
}
```

Note

The same method also applies to event delay calibration.

3.3 PWM and GPIO Test Result

Figure 3-3 shows the test result of PWM and GPIO method.

The phase difference can be controlled by adjusting the CC value of each timer:

- Timer G14 CC1 controls the *pd* value, which is the phase difference between *PWM0* and *PWM1*.
- Timer G14 CC2 controls the *pd0* value, which is fixed at 180°, phase difference between *PWM0+* and *PWM0-*.
- Timer A0 CC1 controls the *pd1* value, which is fixed at 180°, phase difference between *PWM1+* and *PWM1-*.

The duty cycle can be controlled by adjusting the CC value of each timer's CC0:

- Timer G14 CC0 and Timer G8 CC1 control the *dc0*, which is *PWM0+* and *PWM0-*'s duty cycle.
- Timer A0 CC0, A0 CC1, and A0 CC2 control the *dc1*, which is *PWM1+* and *PWM1-*'s duty cycle.

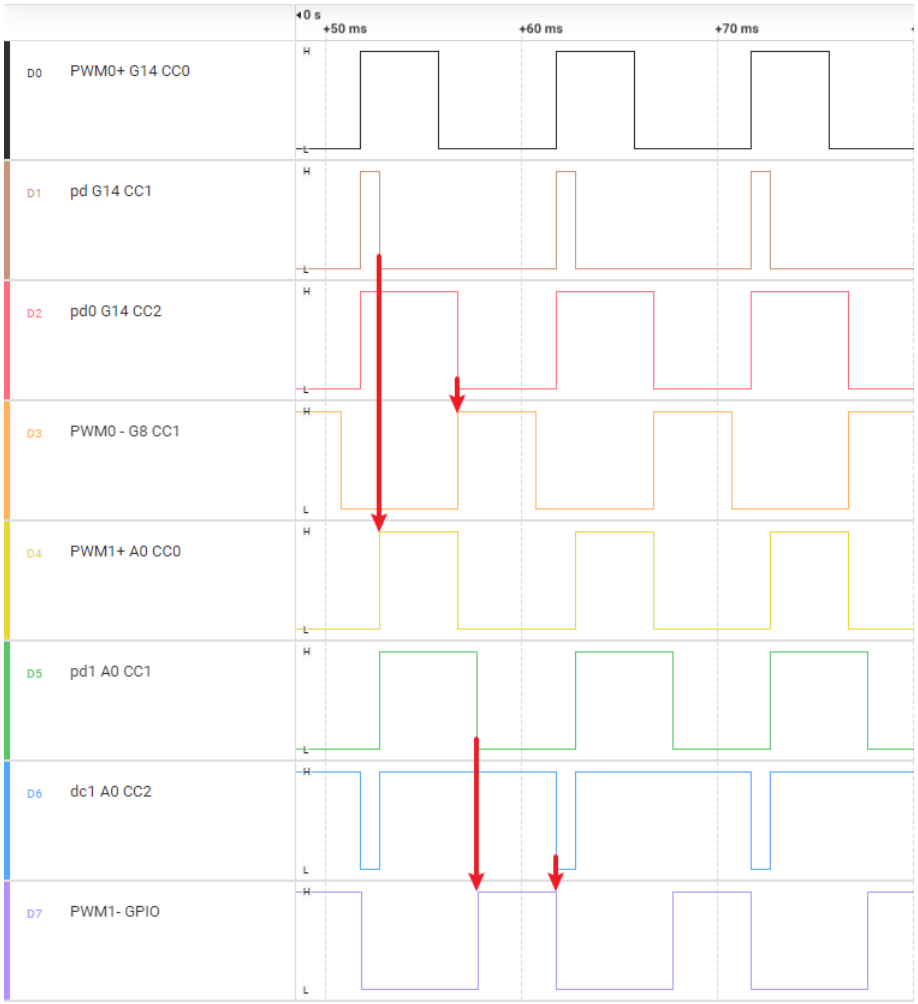

Figure 3-3. PWM and GPIO Method Test Result

4 Timer and GPIO

4.1 Timer and GPIO Implementation

The timer and GPIO method is a software method, and can be used for application that do not need PWM accuracy. Timer interrupt is used to control GPIO set and clear. Using four timers to control four GPIO is better, as MSPM0 can only enter eight interrupt times during each PWM cycle.

Figure 4-1 shows the internal software control chain of timers and GPIOs of this method. This application uses the cross trigger function of the timer to start four timers synchronously.

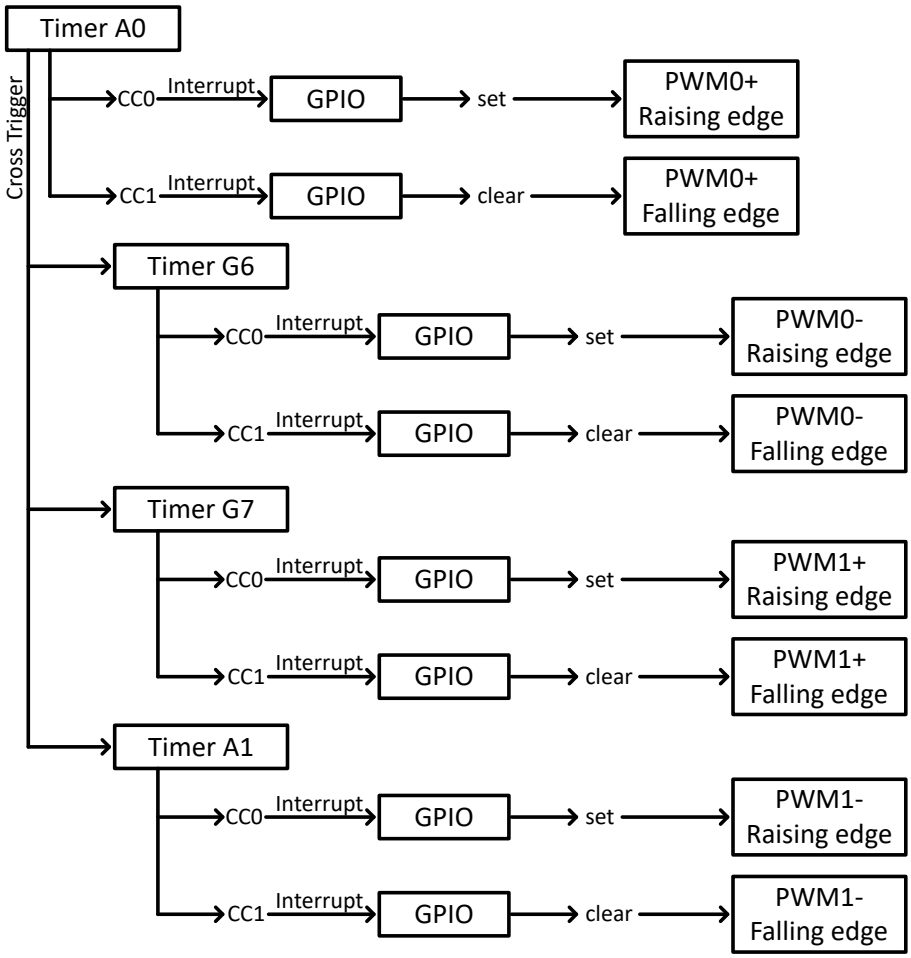

Figure 4-1. Timer and GPIO Method Control Chain

5 Summary

This application note introduces the basic information of LSR and provides a method to reduce speckle on the optical path. Based on the hardware feature of MSPM0, three LSR methods are introduced, as well as the hardware requirements of these methods for MSPM0.

The implementation of three methods and the control chain of MSPM0 are introduced in this document. The PWM method is a hardware method, using an event to control the start of the timer. The PWM and GPIO method is a hardware and software mix control method, which enables the LSR function in MSPM0 C-series devices. The timer and GPIO method is a software method.

6 References

- Texas Instruments, MSPM0 G-Series 80MHz Microcontrollers Technical Reference Manual, technical reference manual
- Texas Instruments, MSPM0 L-Series 80MHz Microcontrollers Technical Reference Manual, technical reference manual
- Texas Instruments, MSPM0 C-Series 80MHz Microcontrollers Technical Reference Manual, technical reference manual
- Texas Instruments, MSPM0G350x Mixed-Signal Microcontrollers With CAN-FD Interface, data sheet
- Texas Instruments, MSPM0L130x Mixed-Signal Microcontrollers, data sheet
- Texas Instruments, MSPM0C110x, MSPS003 Mixed-Signal Microcontrollers, data sheet

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265