Application Note

Dual Wheel Control Design in Projector Based on MSPM0 MCU

TEXAS INSTRUMENTS

Helic Chi, Rocky Chen

ABSTRACT

A dual color wheel control system requires precise synchronization of color wheel position and SoC control signal. The low-cost and low-power microcontroller MSPM0 offer a wide range of peripherals to meet the control requirements of the dual color wheel control system. This application note is a guide for implementation of color wheel position synchronization system based on MSPM0, including the method to implement the software and method to test system performance. This document helps customers to quickly setup and evaluate the system, shorten development cycles and accelerate time-to-market.

Table of Contents

Trademarks

All trademarks are the property of their respective owners.

1 Introduction

In projector, there are several light source technologies. Among them, single color laser is a projector light source technology with high brightness, wide color gamut and long life. Single color laser technology use blue laser to excite the fluorescent material, to get red, green and blue colors first. The laser excites the three-color laser throughout the fluorescent wheel, and then through the synchronous processing of the dispersion wheel, projected onto the DMD.

In the projection optical system, the color excitation of the fluorescent wheel, the color processing of the dispersion wheel, and the reflector control of the DMD chip are all carried out synchronously. Therefore, MSPM0 is required to control the synchronous rotation of the color wheel according to the external synchronization signal.

1.1 Dual Wheel Projector System

In a dual color wheel projector system, DLPC controls both the DMD and the color wheel synchronous signal. MSPM0 is controlled by FW/DW synchronous signal and controls the motor driver speed signal to synchronously control the color wheel rotation position. There are two color wheel in the system: one is fluorescent wheel and the other is dispersion wheel.

Figure 1-1 is a simple diagram of the MSPM0 in a dual color wheel projector system.

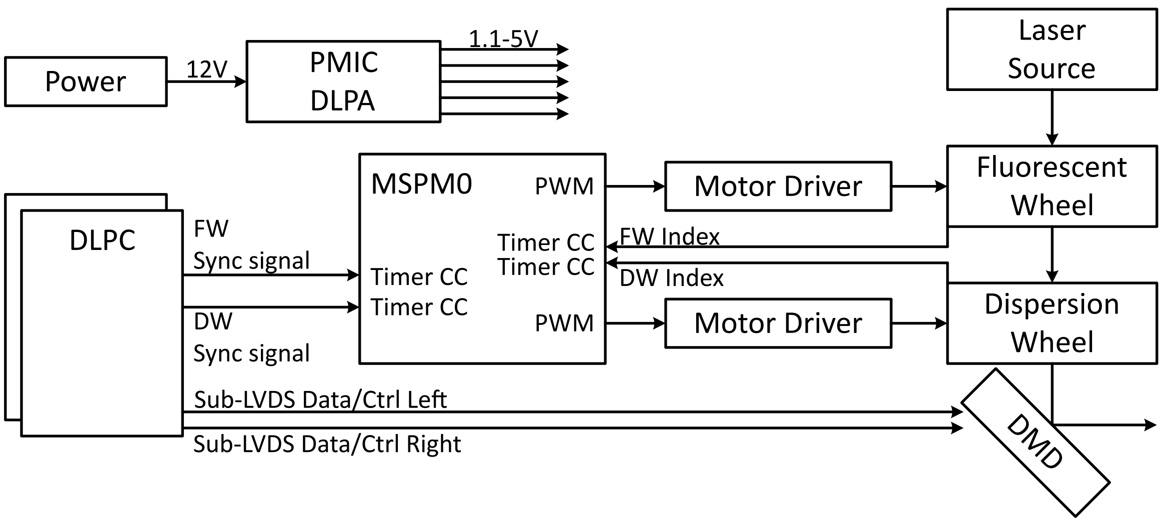

Figure 1-1. Dual Wheel Projector System Diagram

1.2 MSPM0 Requirements

In projector dual wheel control applications, MSPM0 controls the position synchronization of the color wheel by outputting speed control signal to the MCF8316.

Figure 1-2 is the control diagram of MSPM0 in a wheel control application. The MSPM0 uses the same method to control two wheels. The FW and DW Index is the color wheel position feedback signal, when the wheel rotates to the set position, the position detection module trigger a pulse output. FW and DW Sync Signal is the input synchronous signal from DLPC, this is also a pulse signal. MSPM0's timer can calculate the two pulse signal's edge position error by using the hardware capture feature. The PWM speed signal is controlled through the phase loop calibration algorithm module, thereby controlling the speed of the color wheel to make sure that the phase difference between the FW/DW Index pulse edge and FW/DW sync pulse edge is fixed.

Optionally, MSPM0 can capture the sync signal and MCF8316's feedback PWM signal's duty and period information by the capture function of the timer.

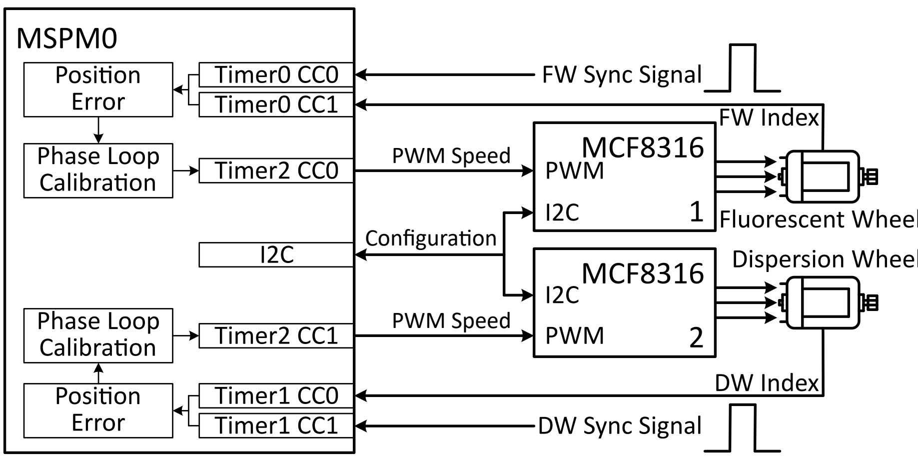

Figure 1-2. MSPM0 Wheel Control Diagram

Based on the above requirements, wheel control application requires at least three timers to control the two color wheels. Table 1-1 is dual wheel control MSPM0 resource requirement.

Table 1-1. Dual Wheel Control MSPM0 Resource Requirement

Function	Peripherals	Description
MCF8316 Configuration	I2C bus	MCF8316 need to configure to different address to share the same I2C bus
Edge difference capture	Timer0 CC0 and CC1	Each color wheel needs a timer with two capture channels
	Timer1 CC0 and CC1	
Speed control	Timer2 CC0 and CC1	Two MCF8316s can use a single timer with two compare channel to output PWM
Phase loop calibration	CPU and MATHACL	Calibration loop algorithm
Signal input filter	COMP	Optional, convert the analog position signal to digital signal
Period and duty capture	Timer3 CC0 and CC1	Optional, capture the color wheel speed, speed of the control signal and duty
Task scheduler	Systick	Periodic calibration task

Note

For edge capture function, if input signal is an analog signal, a MSPM0 internal comparator can be added to the signal chain to convert the analog signal to digital signal. MSPM0 internal comparator has a build-in reference 8-bit DAC that can be used for logic level control, also hysteresis and filter feature can be used for analog signal filter and output a stable digital signal to timer.

2 Color Wheel Control in MSPM0

This section introduces the software structure and peripherals configuration, including timer compare and capture mode configuration, comparator filter configuration, and using MATHACL to accelerate the calibration calculation.

2.1 Software Structure

Figure 1-2 shows MSPM0 internal software structure. There are three necessary software modules in dual wheel control applications, external edge difference capture (Timer0 CC0 and CC1 module), internal position loop control (Phase loop calibration module), and speed control output function (Timer2 CC0).

Also, there are optional software modules, such as synchronous signal period capture, and index signal period capture. These two optional functions are both used as external PWM signal period capture, to give the color wheel feedback information to SoC. If there are requirements, and there are extra timer resources in the MSPM0 that a user is not using, these functions can be added into the system, and used as a feedback or system information capture function for SoC.

2.2 External Edge Difference Capture

MSPM0 can use one timer's two capture channels to capture the external pulse width. By configuring the capture action, MSPM0 can simply capture the edge difference of two pulse signal.

There are many methods to capture the external signal edge difference by configuring the capture action in different ways. Here is one example method.

Firstly, confirm the system requirements from dual wheel control applications, which time interval between two signal's edge need to capture. For example, M0 must capture the time difference between the falling edge of *Sync Signal* and the rising edge of *Index* in the Figure 1-2.

The user must configure the register of the timer to control the timer action at two input signal edges. Table 2-1 are the necessary timer configuration and related register description. Here, CC0 input is configured as the input capture channel of *Index* and CC1 input is configured as the input capture channel of *Sync Signal*. Timer's counter is in down count mode and is reloaded at the rising edge of *Sync Signal*.

Table 2-1. Edge Difference Capture Timer Configuration

Configuration	Register	Description
Syscfg - Timer count mode	CTRCTL.REPEAT	Enable repeat mode to detect edge difference continuously
Syscfg - Timer period	LOAD	Control maximum detection period, if color wheel speed is 100Hz and period is 10ms, set load value greater than 12ms
Syscfg - CC0 capture condition	CCCTL_01.CCOND	Set CC0 capture condition to be triggered by CC0 input rising edge
Syscfg - CC1 capture condition	CCCTL_01.CCOND	Set CC1 capture condition to be triggered by CC1 input falling edge
CC1 load condition	CCCTL_01.LCOND	Set CC1/counter load condition to be triggered by CC1 input rising edge
Counter load control	CTRCTL.CLC	

After enabling edge difference capture timer, then M0 can read the CC0 and CC1 value in the interrupt of CC0 and CC1, the value difference between CC0 and CC1 is the edge difference value. The entire capture process is based on the capture hardware of the timer, and the edge difference cannot be affected by the software execution speed. Then, input captured difference value to calibration module to run the control algorithm.

Note

Based on different CC input signal edge capture requirement and waveform, there are many configuration to setup hardware edge difference capture function by using the timer of the MSPM0. At the same time, the *Index* and *Sync Signal* are not limited to being input into CC0 and CC1 in a fixed order, and the advanced timer in MSPM0 also supports CC2 and CC3 as capture input. See the MSPM0 device's datasheet and technical reference manual for detailed information.

Note

Users can easily modify a timer to edge difference capture function based on MSPM0-SDK driverlib timer's example code by setting a capture mode to multiple capture.

2.2.1 Input Filter with Comparator

For those application using an analog signal as *Index* or *Sync Signal*, analog signal is not designed for directly input to Timer's CC input, since Timer doesn't filter the analog signal. Or timer's capture function outputs a unstable output, causing color wheel control unstable.

By using MSPM0's build-in COMP(comparator peripheral), MSPM0 can simply filter the input analog signal.

Also, there is a 8-bit DAC interated in COMP, MSPM0 can also set a reference voltage, help application easily convert analog signal to digital signal with customizable convert thresholds.

For COMP's hysteresis feature, set a desired hysteresis voltage value, COMP can filter the input analog signal and output a stable digital signal to capture and calibration module, this keeps entire control system running without small disturbances.

Table 2-2 shows the key features of COMP that can be applied in color wheel control application.

Table 2-2. COMP Feature in Color Wheel Control Application

Configuration	Register	Description
Syscfg - Input channel	CTL0.IPSEL	Enable positive input channel for nanlog signal input
Syscfg - Reference voltage generator	CTL2.REFSRC	VDDA selected as the reference source to DAC and DAC output applied as reference to comparator
Syscfg - Hysteresis	CTL1.HYST	Hysteresis setting of the comparator
Syscfg - Output filter	CTL1.FLTENCTL1.FLTDLY	Enable the output filter

Note

Users can simply modify a timer to edge difference capture function based on MSPM0-SDK driverlib comp's example code by adding a filter, hysteresis and 8-bit DAC as reference feature in syscfg.

2.3 Internal Position Loop Control

Position loop control algorithm is running on the CPU of the MSPM0. Figure 2-1 is the MSPM0 PID loop control block diagram. In the color wheel application, MSPM0 uses PI control to calculate and calibrate the duty error based on input edge difference captured by the timer. The output is a offset of the duty of MCF8316's PWM speed control signal.

If a phase offset is needed, make sure this offset is added to Phase Setting in Figure 2-1.

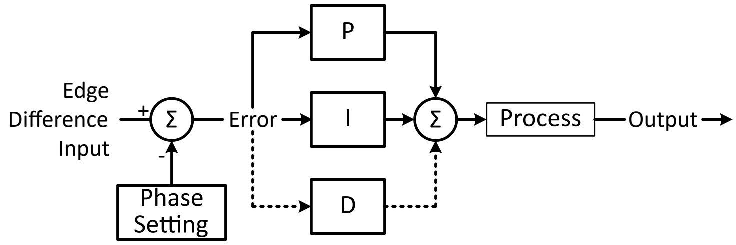

Figure 2-1. Block Diagram of MSPM0 PID control

Note

The P, I parameters of the control loop need to be adjusted by the user based on experience and the form of the output parameters.

2.4 Speed Control Output

For the front-end analog motor driver, PWM is used as speed control output from MSPM0 to the motor driver. Usually, the speed reference for the motor driver is the duty cycle of PWM.

The center speed of the color wheel is constant under a fixed configuration, but sync signal from SoC changes depending on the image rendering frequency and these two asynchronous clock systems. Therefore, M0 needs to control the motor to adjust the position of the color wheel in real time around the center rotational speed. Therefore, based on this speed control analysis, speed offset output from the Internal Position Loop Control is added to the constant center speed of the color wheel -duty cycle of the PWM.

If a 32-bit timer is available on the target MSPM0 device, the 32-bit timer can output high-precision PWM without timer input clock frequency division, which plays an important role in overall control accuracy.

Note

Users can setup a PWM output function based on MSPM0-SDK driverlib PWM's example.

3 Test and Evaluation Method

3.1 Test System Setup

Users can setup customized test systemd based on available hardware and project requirements. However, some key components are still required, by checking the system diagram Figure 1-1 and listed below.

- A PWM generator to input a sync signal for MSPM0. DLPC is this PWM source, but in development phase, a MSPM0 or signal generator is OK.
- A MSPM0 EVM, LP-MSPM0G3507 is recommended.
- A motor driver. For example, MCF8316, part number depends on the motor type.
- A color wheel motor with position sensor. Output signal of the position sensor is crucial, which determines how the MSPM0's PWM capture function is configured and position capture accuracy.

Next step is to setup the MSPM0 software project. Color Wheel Control in MSPM0 introduces MSPM0 software structure in color wheel control application.

To increase the MSPM0 output PWM frequency and duty cycle accuracy, select MSPM0 G-series device and set MCLK to 80MHz.

3.2 Performance Evaluation

After finishing the implementation of each module and cascading of modules, the next step is to adjust the loop control parameters, and optimize the internal position loop control performance, making sure that the phase difference between the motor position feedback signal (*Index*) edge and the SOC synchronization signal (*Sync*) edge is stable.

One recommended phase stabilization test method is to use an oscilloscope to monitor two signal's edge difference with afterglow enabled. The color wheel application uses the *Sync* signal as a reference signal to test the tracking stability of the color wheel position - *Index* signal. Therefore, TI recommends setting the *Sync*'s edge as the trigger of the oscilloscope to observe the tracking performance and stability of the color wheel position - *Index* signal relative to the *Sync*.

4 Summary

This application note introduces a method on how to use MSPM0 to setup a dual wheel control system in laser projector systems. First, briefly introduce the architecture and functional requirements of the color wheel system; then, this document introduces how to use MSPM0 to build the system control loop; finally, the system setup method and performance evaluation method are given.

5 References

- Texas Instruments, MSPM0 G-Series 80MHz Microcontrollers Technical Reference Manual, technical reference manual.
- Texas Instruments, MSPM0G350x Mixed-Signal Microcontrollers With CAN-FD Interface, data sheet.
- Texas Instruments, MCF8316A Sensorless Field Oriented Control (FOC) Integrated FET BLDC Driver, data sheet.
- Texas Instruments, MSPM0G3507 LaunchPad Development Kit User's Guide, EVM user's guide.
- Texas Instruments, DLP controllers & drivers, product selection page.