

Subsystem Design

LED Driver With PWM

Description

The PWM duty cycle directly correlates to the brightness of the LED. When using an LED as an indicator or a light source in an application, you can use a PWM signal to drive the LED brightness and power consumption. The timer modules in the MSPM0 can be used to generate PWM signals with varying frequency and duty cycles. This example code dims and brightens the LED in a heartbeat manner to display the full range of PWM duty cycles that can be used to drive an LED.

Figure 1-1 displays a functional block diagram of the peripherals used in this example.

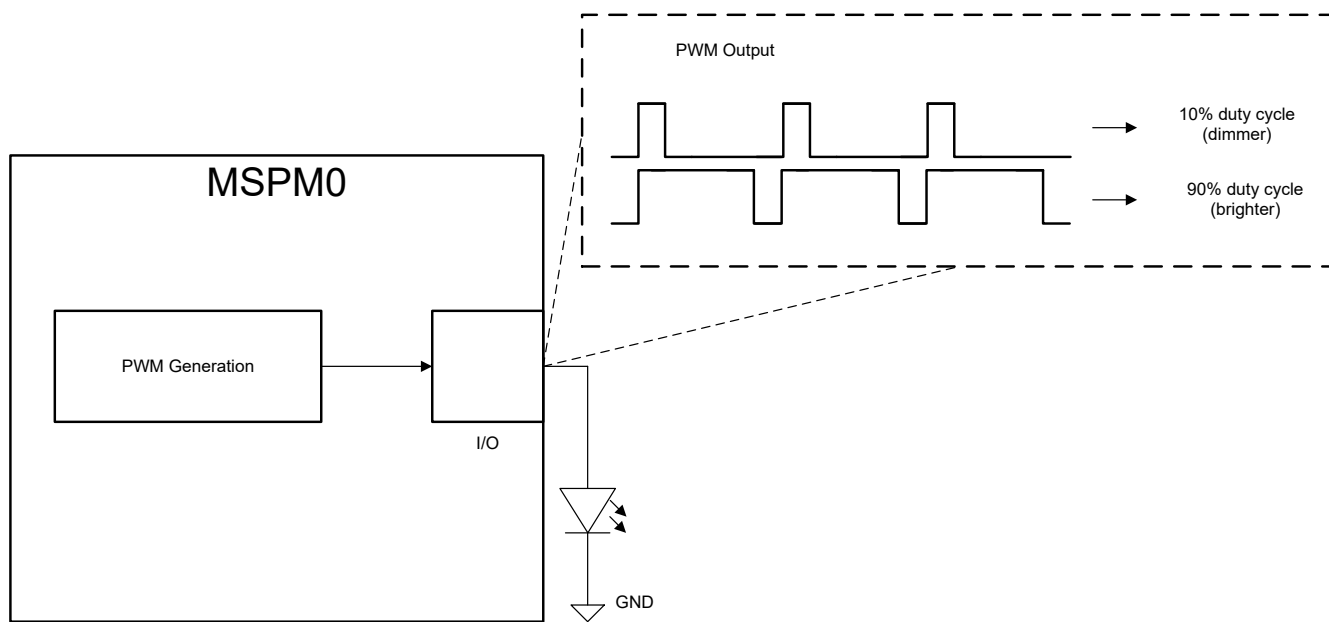


Figure 1-1. Subsystem Functional Block Diagram

Required peripherals

This application requires one timer, one device pin, and an onboard LED.

Table 1-1.

Sub-block Functionality	Peripheral Use	Notes
PWM generation	(1x) Timer G	Called PWM_0_INST in code
IOMUX sub-block	1 pin	(1x) PWM output

Compatible devices

Based on the requirements in [Table 1-1](#), this example is compatible with the devices in [Table 1-2](#). The corresponding EVM may be used for prototyping.

Table 1-2.

Compatible Devices	EVM
MSPM0Lxxx	LP-MSPM0L1306
MSPM0Gxxx	LP-MSPM0G3507

Design steps

1. Determine the required PWM output frequency and resolution. These two parameters will be the starting point when calculating other design parameters; the frequency should be determined by how quickly the status of an external component needs to be updated. In this example, we chose a PWM output frequency of 62 Hz and a PWM resolution of 2000 bits.
2. Calculate the timer clock frequency. The following equation can be used to calculate the timer clock frequency: $F_{\text{clock}} = F_{\text{pwm}} \times \text{resolution}$
3. Configure peripherals in [SysConfig](#). Select which timer instances are used and which device pins are used for PWM output. This example uses PA13 for the PWM output (which is connected to TIMG0).
4. Write application code. The remaining piece of this application is to change the PWM duty cycle, which is accomplished in software. See [Figure 1-2](#) for an overview of the application or browse the code directly.

Design considerations

1. Maximum output frequency: Fundamentally, the max PWM output frequency is limited by both the IO speed and selected clock source frequency. However, the duty cycle resolution also affects the max output frequency. More resolution requires more timer counts which increases the output period.
2. Pipelining: The PWM timer selected in this application supports pipelining the timer compare value. Pipelining allows the application to schedule an update to the timer compare value without causing a glitch on the output.

Software flowchart

[Figure 1-2](#) shows the operations performed by application to change the duty cycle of the PWM output.

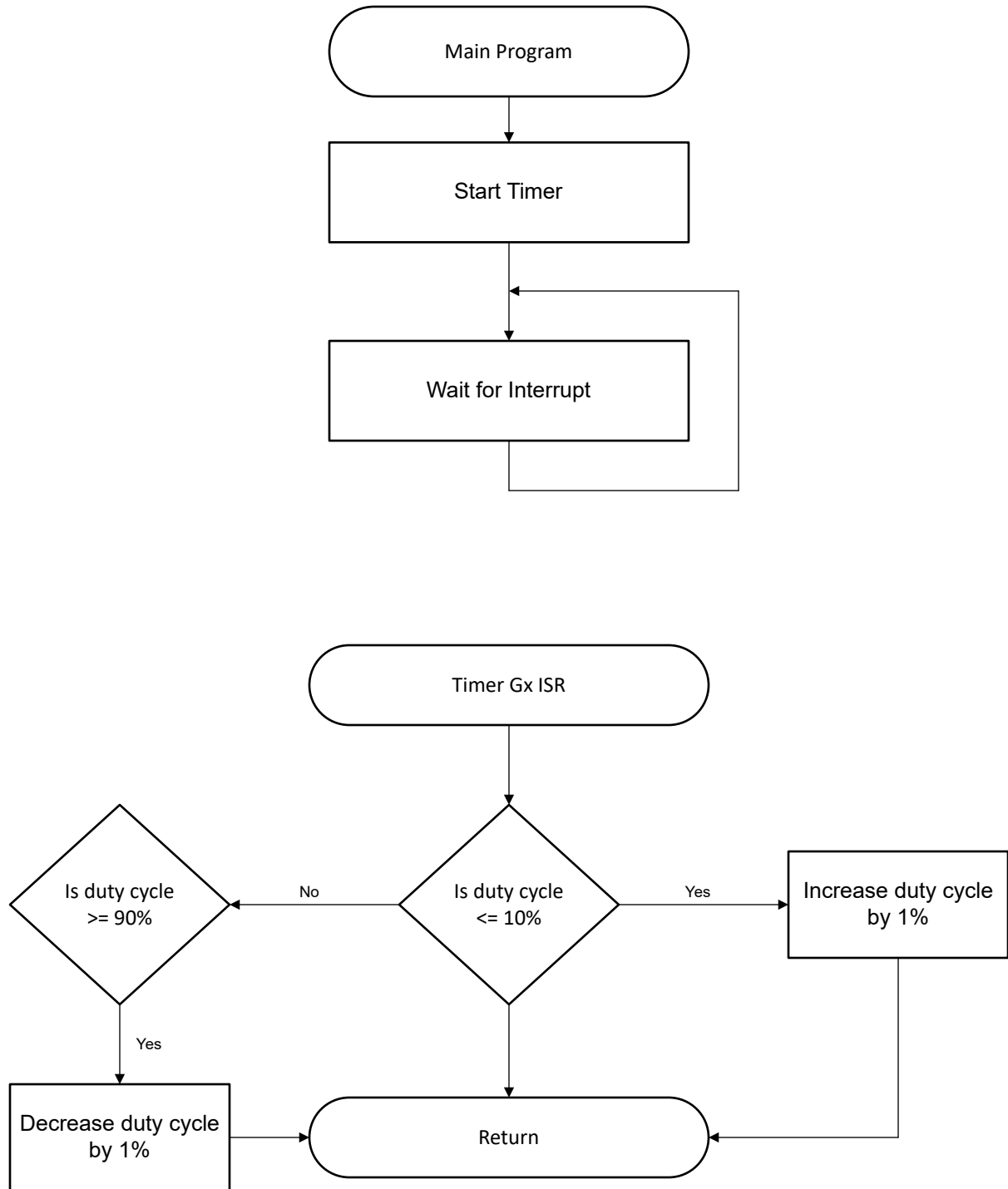


Figure 1-2. Application Software Flowchart

Application code

In the application code, the PWM duty cycle is increased by 1% each time the timer triggers an interrupt until it reaches 90% and then decreased by 1% until the duty cycle reaches 10%, which generates a heartbeat effect. This application PWM output has 2000 bits of resolution; therefore, increasing or decreasing the *pwm_count* variable by 20 changes the duty cycle by 1%. Depending on an application requirements, different scaling can be required.

```
void PWM_0_INST_IRQHandler(void){
    switch (DL_TimerG_getPendingInterrupt(PWM_0_INST)){
        case DL_TIMER_IIDX_LOAD:
            if (dc <= 10){mode = 1;} // if reached lowest dc (10%), increase dc
            else if (dc >= 90){mode = 0;} // if reached highest dc (90%), decrease dc
            if (mode){pwm_count -= 20; dc += 1;} // up
            if (!mode){pwm_count += 20; dc -= 1;} // down
            DL_TimerG_setCaptureCompareValue(PWM_0_INST, pwm_count, DL_TIMER_CC_1_INDEX); // update
        ccr1 value
            break;
        default:
            break;
    }
}
```

Results

Additional Resources

- [Download the MSPM0 SDK](#)
- [Learn more about SysConfig](#)
- [MSPM0L LaunchPad development kit](#)
- [MSPM0G LaunchPad development kit](#)
- [MSPM0 Timer PWM academy](#)

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](https://www.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
Copyright © 2024, Texas Instruments Incorporated