Using 4 PWMs to Control Half-bridge DC/DC Circuit

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	This sample code uses M480 series microcontroller (MCU) outputting 4 PWM waveforms to control half-bridge DC/DC circuit.
BSP Version	M480 BSP CMSIS V3.05.001
Hardware	NuMaker-M483KG Ver1.1

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

Half-bridge DC/DC circuit uses four PWMs. Two of the PWMs drive transformer's primary coil, and the other two PWMs are used for synchronous rectification of secondary coil. When DC/DC works normally, SR1 and SR2 are usually both turned on while the primary coil has no electric current as Figure 1-1.

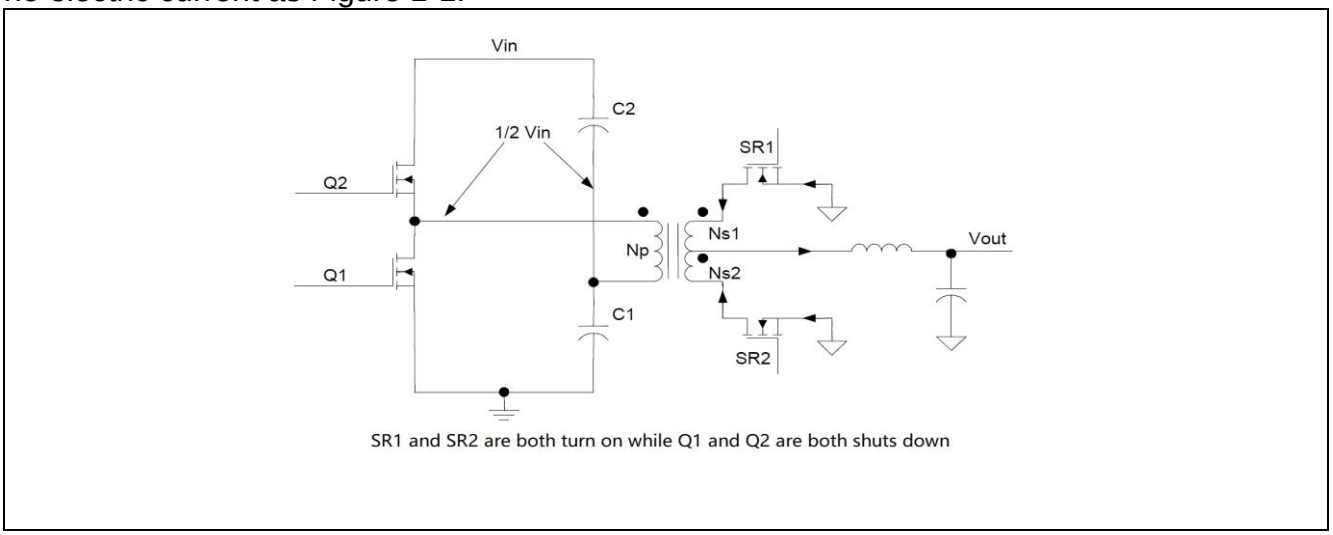

Figure 1-1 Half-bridge DC/DC Circuit

SR1 shuts down prior to Q2 turn-on. Thus, SR1 and Q2 can use a pair of complementary PWMs. As the same reason, Q1 and SR2 can use a pair of complementary PWMs. Electric current direction is shown as Figure 1-2.

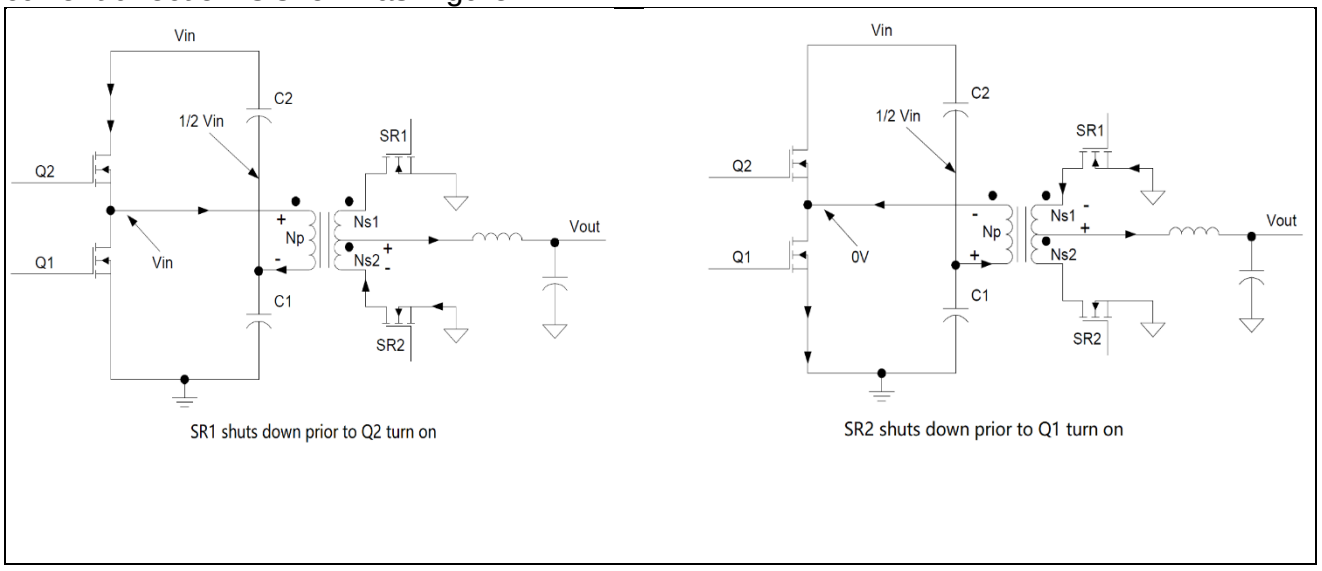

Figure 1-2 Electric Current Direction

The primary coil voltage waveform is shown as Figure 1-3. The turn-on phase between Q1 and Q2 must be 180 degrees.

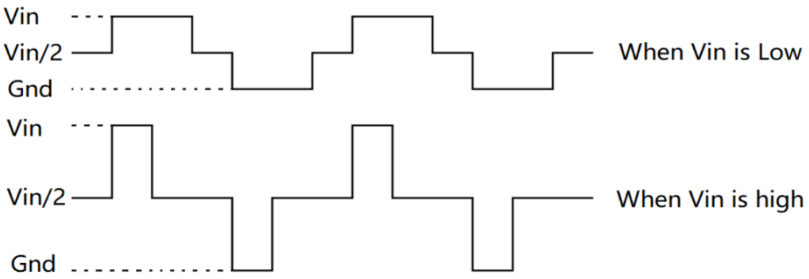

Figure 1-3 Primary Voltage Waveform

1.1 Demo Result

Figure 1-4 shows an example waveform of four PWMs. Yellow is PWM0 driving Q1 and green is PWM2 driving Q2. The phase between PWM0 and PWM2 is 180 degrees. The 3rd line, red, is PWM1 driving SR2 for synchronous rectification. These waveforms are just the executive result of code.

Figure 1-4 Waveforms Output from PWM0,PWM2 and PWM1

2 Code Description

This example code configures PWM0 outputting 1 at zero and PWM2 outputting 1 at center point as Figure 2-1. The phase between PWM0 waveform and PWM2 is just 180 degrees.

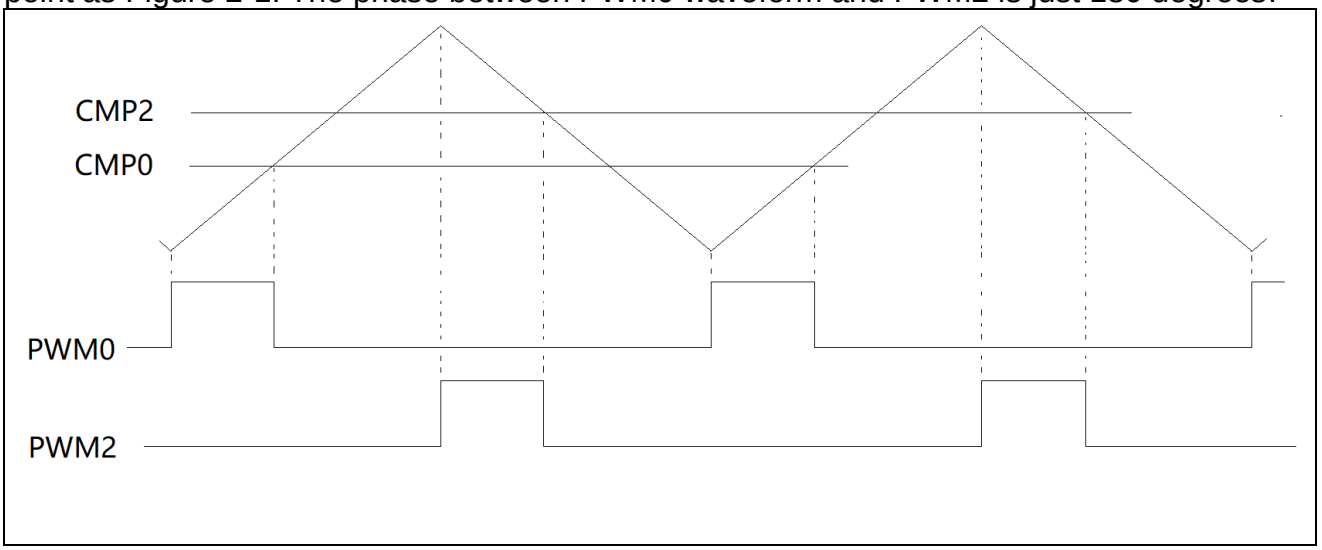

Figure 2-1 PWM0 and PWM2 Waveforms

PWM0~3 must be configured to start synchronously in initialization subroutine.

```
    EPWM0->CTL1   = 0x03000000       // PWM01,23 is complementary mode
                  | 0x000000AA ;     // PWM01,23 Up-down counter type

    EPWM0->WGCTL0 = 0x00000002       // PWM0 Zero point output 1
                  | 0x00010000       // PWM0 center point output 0
                  | 0x00000010       // PWM2 Zero point output 0
                  | 0x00200000 ;     // PWM2 center point output 1

    EPWM0->WGCTL1 = 0x00000001       // PWM0 CMP UP output 0
                  | 0x00100000 ;     // PWM2 CMP DOWN output 0

    EPWM0->PERIOD[0] = 1000 ;        // PWM frequency =192M/1000/2 =96KHz
    EPWM0->CMPDAT[0] = 0 ;
    EPWM0->PERIOD[2] = 1000 ;
    EPWM0->CMPDAT[2] = 0 ;           // Duty = 0

    EPWM0->DTCTL[0] = 0x00010000     // Enable DeadZone, Count EPWM_CLK
                    | 96 ;           // Dead time = 96/192M = 0.5us
    EPWM0->DTCTL[1] = 0x00010000     // Enable DeadZone, Count EPWM_CLK
                    | 96 ;           // Dead time = 96/192M = 0.5us

    EPWM0->CNTCLR = ~0 ;
    EPWM0->IFA[0] = 0 ;              // At first,it MUST be writed 0
    EPWM0->IFA[0] = 0x80000000       // Enable interrupt accumulator
                  | 1 ;              // interrupt every (Num+1) period

    EPWM0->AINTSTS = ~0 ;            // Clear interrupt flag
    EPWM0->AINTEN = 1 ;

    /* Lock protected registers */
    SYS_LockReg();

    NVIC_SetPriority(EPWM0P0_IRQn, 2) ;
```

```
    NVIC_ClearPendingIRQ(EPWM0P0_IRQn);
    NVIC_EnableIRQ(EPWM0P0_IRQn);                 // Enable PWM0-CH0 interrupt

    /* Enable counter synchronous start function for EPWM0 channel 0~3 */
    EPWM_ENABLE_TIMER_SYNC(EPWM0, 0x0F, EPWM_SSCTL_SSRC_EPWM0);

    /* Trigger EPWM counter synchronous start by EPWM0 */
    EPWM_TRIGGER_SYNC_START(EPWM0);

    /* Enable output of EPWM0 channel 0~3*/
    EPWM_EnableOutput(EPWM0, 0x0F);
```

Duty must be adjusted every period to stabilize output voltage, which is done in the PWM interrupt subroutine.
Duty or period will be adjusted according to output voltage in actual application. How to adjust duty or period is demonstrated in the interrupt subroutine.

```
// Period and duty can be changed every PWM interrupt
//
void EPWM0P0_IRQHandler(void)
{
    int32_t static cnt = 0 ;

    if (cnt)
    {
        EPWM0->PERIOD[0] = 2000 ;     // change Period
        EPWM0->PERIOD[2] = 2000 ;
        EPWM0->CMPDAT[0] = 1600 ;     // change duty
        EPWM0->CMPDAT[2] = 400 ;
    }
    else
    {
        EPWM0->PERIOD[0] = 1000 ;
        EPWM0->PERIOD[2] = 1000 ;
        EPWM0->CMPDAT[0] = 300 ;
        EPWM0->CMPDAT[2] = 700 ;
    }

    EPWM0->AINTSTS = ~0 ;             // Clear Flag of interrupt
    cnt ^= 3 ;
}
```

3 Software and Hardware Requirements

3.1 Software Requirements

- BSP version
  - M480_BSP_CMSIS_V3.05.001
- IDE version
  - Keil uVersion 5.26

3.2 Hardware Requirements

- Circuit components
  - NuMaker-M483KG V1.1
- Pin Connect

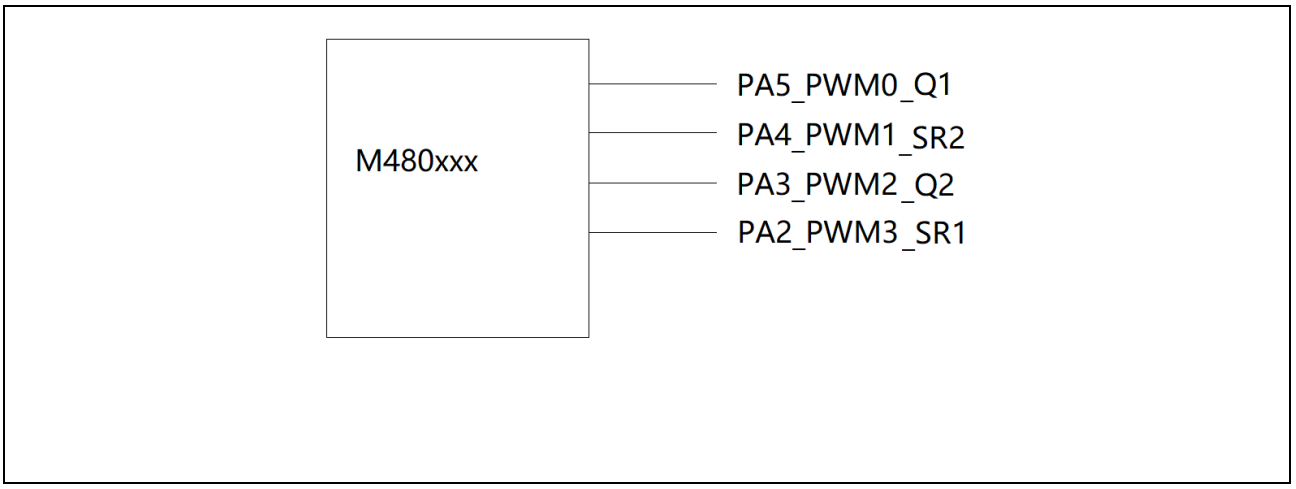

Figure 3-1 Pin Connect

4 Directory Information

The directory structure is shown below.

EC_M480_4PWM_for_Half-bridge_DCDC_V1.00	
Library	Sample code header and source files
CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
SampleCode	
ExampleCode	Source file of example code

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 *M480_4PWM_for_Half-Bridge_DCDC.uvproj*.
2. Enter Keil compile mode.
   - Build
   - Download
   - Start/Stop debug session
3. Enter debug mode.
   - Run

6 Revision History

Date	Revision	Description
2021.12.01	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.