AN-1407
Application Note

One Technology Way • P.O. Box 9106 • Norwood, MA 02062-9106, U.S.A. • Tel: 781.329.4700 • Fax: 781.461.3113 • www.analog.com

ADSP-CM402F/ADSP-CM403F/ADSP-CM407F/ADSP-CM408F/ADSP-CM409F Pulse Width Modulator in AC Motor Control Applications

by Dara O'Sullivan, Jens Sorensen, and Aengus Murray

INTRODUCTION

This application note introduces the main features of the ADSP-CM402F/ADSP-CM403F/ADSP-CM407F/ADSP-CM408F/ADSP-CM409F pulse width modulator (PWM) and use in 3-phase ac motor control applications. The PWM peripheral is capable of driving inverters for a variety of power converter applications, including standard 3-phase ac inverters, multilevel ac inverters, and a variety of dc-to-dc converters. There are three PWM peripheral blocks, each with four pairs of PWM outputs. The controller supports all ac motor types and includes features that support six-step control of brushless dc (BLDC) motors and control of switched reluctance motors. This application note focuses on 3-phase ac inverter control. For more information about the full range of PWM controller features and configuration registers, see the ADSP-CM40x Mixed-Signal Control Processor with ARM Cortex-M4 Hardware Reference Manual and the documentation within the ADSP-CM40x Enablement Package software.

3-PHASE MOTOR CONTROL

The 3-phase voltage fed inverter, shown in Figure 2, drives a 3-phase ac motor under the control of a PWM modulator connected to a microcontroller central processing unit (CPU). The inverter produces fixed frequency variable duty cycle waveforms modulated with the target motor voltage and frequency. The motor windings filter the high frequency components and motor current is at the fundamental frequency with some residual ripple. The inverter behaves as a variable frequency ac source with outputs ranging from 0 to V_{DC}, centered around $V_{DC}/2$. The CPU executes a digital control algorithm that calculates required inverter voltages at the switching frequency. Typically, the control algorithm also requires winding current feedback and the PWM modulator provides synchronizing triggers to the CPU, analog-to-digital converter (ADC), and other microcontroller peripherals.

The inverter waveform, shown in Figure 1, is a center-based PWM waveform where the on period grows or shrinks around the midpoint of the switching waveform.

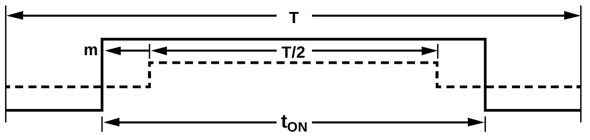

Figure 1. Inverter Switching Waveform

The inverter output as a function of the on time is

$$V = V_{DC}\left(\frac{t_{ON}}{T}\right) = \frac{V_{DC}}{2}\left(\frac{t_{ON} - \frac{T}{2}}{\frac{T}{2}}\right) + \frac{V_{DC}}{2} \quad (1)$$

where t_{ON} is the on time of the PWM output, defined in Figure 2.

When driving a balanced 3-phase load such as an ac motor, setting 50% duty cycle to all three phases applies 0 V across the windings. Therefore, the inverter output of $V_{DC}/2$ at 50% duty cycle corresponds to zero voltage. Then duty cycles greater than 50% produce positive voltages and duty cycles less than 50% produce negative voltages. This ac voltage, V_{AC}, scales according to a modulation function (m). This equation defines the operation of the PWM modulator:

$$V_{AC} = m\frac{V_{DC}}{2} \quad (2)$$

Where m = $\dfrac{t_{ON} - \frac{T}{2}}{\frac{T}{2}}$.

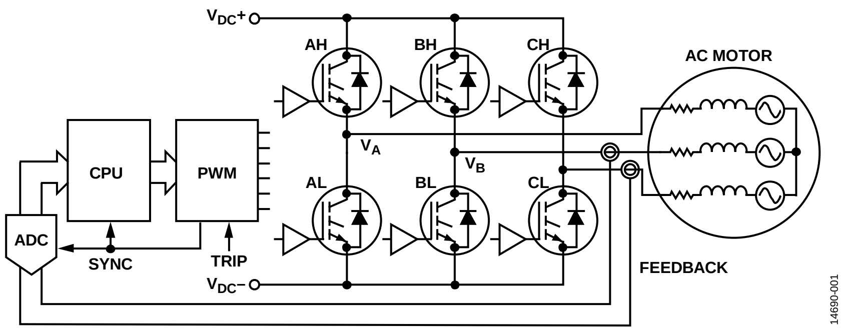

Figure 2. 3-Phase AC Motor Drive

TABLE OF CONTENTS

REVISION HISTORY

8/2016—Revision 0: Initial Version

The PWM modulator controls the inverter by turning on and off the high and low side power transistors to switch the outputs between the high and low side dc bus. The PWM signals driving the power transistor gates must account for the turn on and turn off delays in the power transistors so the PWM modulators insert a dead time between the high and low side gate signals to eliminate the possibility of cross conduction. Dead time insertion on the PWM signals is illustrated in Figure 3.

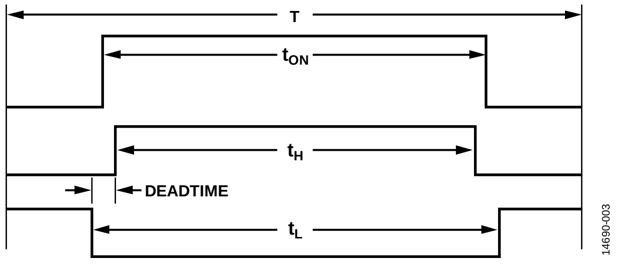

Figure 3. Dead Time Feature

The remaining safety feature required for the PWM modulator is a trip function that turns off all power transistors in the event of an inverter fault. The PWM trip signal derives from both internal and external fault detection circuits and typically bypasses the CPU. The PWM modulator sends an interrupt signal to the CPU on trip events to initiate the fault handling sequence.

PWM MODULATOR OPERATION

Figure 4 shows the important functions of a PWM modulator including the PWM timer, timing control, dead time, and trip control units. The timing control units detect the crossovers between the PWM timer output and the duty cycle references A, B, and C.

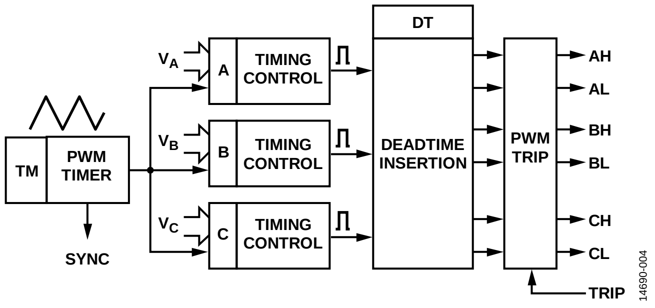

Figure 4. Basic PWM Controller

The PWM timer generates a triangular reference wave, shown in Figure 5, that counts between +TM/2 and −TM/2. The modulator generates a 50% duty cycle output with a zero reference input corresponding to an ac output voltage of zero from the inverter.

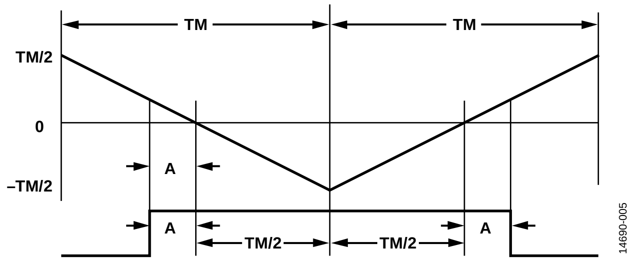

Figure 5. PWM Waveform Generation

By defining the duty cycle reference (M) to run from −TM/2 to +TM/2, the inverter output voltage is found through Equation 1 and Equation 2:

$$V = V_{dc}\left(\frac{t_{on}}{T}\right) = \frac{V_{dc}}{2}\left(\frac{M}{\frac{TM}{2}}\right) + \frac{V_{dc}}{2} \tag{3}$$

The duty cycle reference as a function of the ac voltage component is

$$M = \frac{TM}{2}\left(\frac{V_{ac}}{\frac{V_{dc}}{2}}\right) \tag{4}$$

The timer is clocked by the peripheral system clock, f_{SYSCLK}, and the timer counts through two TM per PWM period (see Figure 5). Consequently, the period measured in peripheral system clock cycles is twice the count value TM, so the PWM switching frequency f_{PWM} is

$$f_{PWM} = \frac{f_{SYNCLK}}{(TM \times 2)} \tag{5}$$

The dead time unit inserts a blanking period, DT, to generate pairs of dead time compensated PWM signals. Independently of the timer operation and the CPU, the PWM trip unit can bring the PWM signals xH and xL (with x = A, B, and C) to a safe state in case of a fault. Typically, trip is connected to an overcurrent detection circuit. The PWM timer also generates a SYNC timing pulse at the start of the PWM cycle to synchronize operation with the motor control algorithm execution and other device peripherals.

PWM CONTROLLER CONFIGURATION

The PWM control peripherals on the ADSP-CM402F/ADSP-CM403F/ADSP-CM407F/ADSP-CM408F/ADSP-CM409F, shown in Figure 6, includes a comprehensive set of functions to support both ac and dc power converter control. The ADSP-CM402F/ADSP-CM403F/ADSP-CM407F/ADSP-CM408F/ADSP-CM409F includes three identical PWM controllers (PWM0, PWM1, and PWM2). The register names with the PWMx_ prefix in the ADSP-CM402F/ADSP-CM403F/ADSP-CM407F/ADSP-CM408F/ADSP-CM409F data sheet and the ADSP-CM40x Mixed-Signal Control Processor with ARM Cortex-M4 Hardware Reference Manual have three instantiations in the IC (x = 0, 1, 2).

The software support files define full register names with the control module name as a prefix.

This section describes configuration settings suitable for 3-phase ac motor control and options for synchronizing the PWM controller with other microcontroller peripherals. The PWM controller has two categories of registers: configuration registers that set operation modes and control registers that define PWM waveform outputs. The control registers are double-buffered registers that the user can load at any time during the PWM cycle. However, the values only take effect at the PWM waveform boundaries. Only load the configuration registers while the PWM controller is disabled. Table 1 shows the list of double-buffered control registers, including timer period registers, channel duty cycle registers, channel control registers, and the dead time register.

For the purposes of this application note, the PWM configuration process splits into four functions:

1. Timer selection and synchronization
2. Waveform mode selection
3. Output control and trip handling
4. Interrupt generation and trigger routing

The rest of the configuration process involves setup of the pin mux that connects the PWM signals to output pins and the trigger routing unit (TRU) that connects PWM triggers to other peripherals. See references to code examples showing the configuration programming in the Appendix section; visit https://ez.analog.com/docs/DOC-12643 for more information.

PWM Timer Configuration and Synchronization

3-phase ac motor control requires the synchronized operation of three channels using a common PWM timer. The common timer is PWMTMR0 and the PWM_TM0 register sets the switching frequency according to the period count, TM, calculated from the following equation:

$$TM = \frac{f_{SYSCLK}}{(f_{PWM} \times 2)} \quad (6)$$

Control bits in the PWM_CHANCFG register select the PWMTMR0 timer as the A, B, and C channel timing input. Control bits in the PWM_CTL register select PWMTMR0 as the PWM_SYNC trigger master to generate interrupts and trigger other peripheral devices on the PWM period boundaries. The sync generation circuits create an extended pulse that is available as an output on the SYNC pin. The contents of the PWM_SYNC_WID register define the width of the PWM_SYNC_OUT pulse.

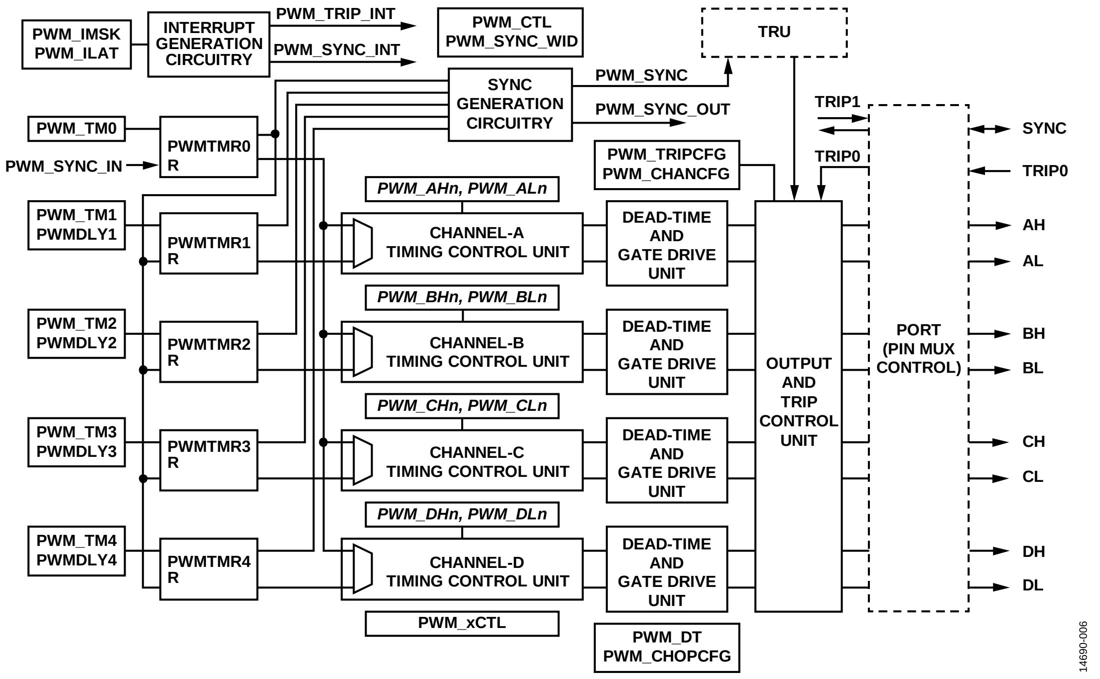

Figure 6. ADSP-CM402F/ADSP-CM403F/ADSP-CM407F/ADSP-CM408F/ADSP-CM409F PWM Control Peripheral

In single axis motor drives, internal signals reset the PWM timer but in multi axis or networked systems, a master controller provides an external trigger. When using internal synchronization, the PWMTMR0 resets on the period boundaries set by the contents of the PWM_TM0 register. When using external synchronization, the external trigger periodically resets the PWMTMR to synchronize the internal clock with the external time base. The recommended operation mode is to synchronize the external trigger with the system clock and to set PWM period, TM, to be an even submultiple of the external trigger clock period as shown in Figure 7. A mismatch in the trigger timing truncates or extends the timing ramp on the external trigger edge, which will cause output voltage jitter.

The PWM_CTL register includes bit settings to select internal or external synchronization and asynchronous or synchronous acquisition of an external trigger, if necessary. The PORT controller, described in the Pin Multiplexer Configuration section, connects the required input or output PWM synchronization signals to the SYNC pin.

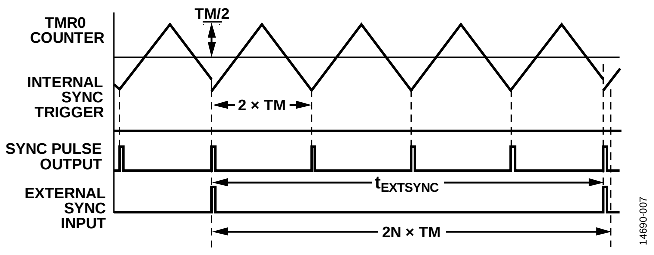

Figure 7. PWM Internal and External Sync Triggers

The GLOBEN bit in the PWM_CTL register begins the operation of the PWM controller. However, this is the last step in the configuration process and left until all PWM, pin multiplexing, and trigger routing configuration settings are complete.

PWM Waveform Mode Selection

3-phase ac motor control typically requires three symmetrical center-based PWM control waveforms, shown in Figure 8. The inverter transistor switching signals are a complementary pair of PWM signals derived from each center-based signal. The complementary waveforms include a period with zero output (dead time) between switching edges.

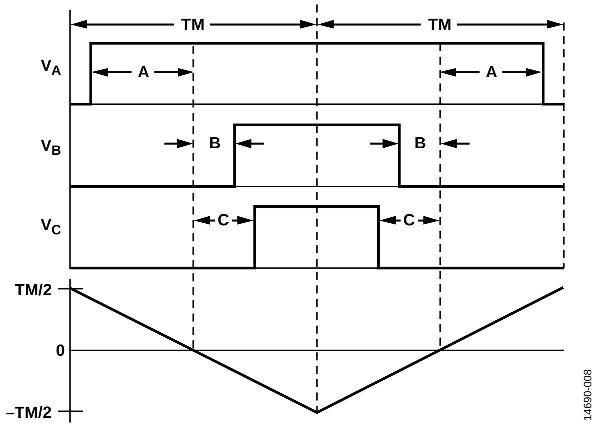

Figure 8. 3-phase Center Based PWM Waveforms

Figure 9 shows the two options for the insertion of this dead time. Symmetrical dead time advances the turn off edge and delays the turn on edge by equal amounts to maintain waveform symmetry about the central axis. Asymmetrical dead time maintains the turn off edges and inserts the full dead time delays before the turn on edges. The dead time between switching edges and the high side and low side are the same for both schemes, but the edge timing relative to the SYNC pulse differs.

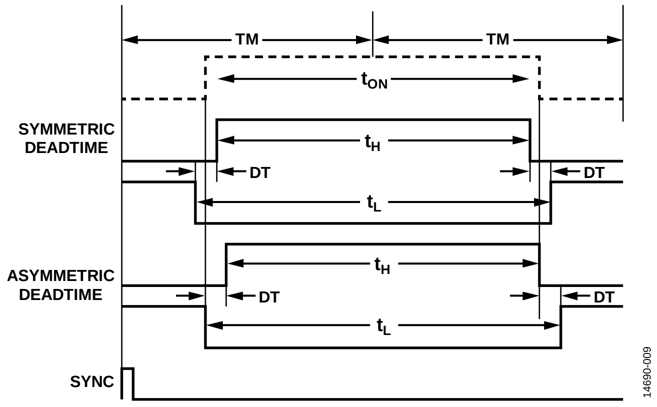

Figure 9. Symmetrical and Asymmetrical Deadtime Insertion.

Using over modulation is a unique operating mode because there is no dead time in a 100% or 0% duty cycle waveform. However, when entering or leaving over modulation the controller inserts a dead time on the rising edge of the high or low side output that switches at the PWM period boundary (see Figure 10. This guarantees the PWM controller always inserts dead time between high side and low side switching edges.

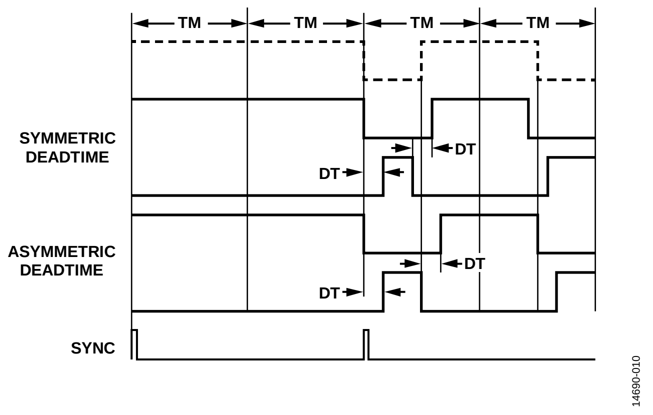

Figure 10. Dead time Insertion Transition from 100% Modulation.

Control bits in the PWM_CHANCFG register select inverted PWM signals on the low side outputs. Control bits in the PWM_CTL register select symmetric or asymmetric dead time insertion and the contents of the PWM_DT register set the dead time delay, according to the following equation:

$$PWM\_DT = \frac{T_{deadtime} \times f_{SYSCLK}}{2} \quad (7)$$

The PWM_CTL register also includes control bits that disable the high and low side outputs and allow the user to turn on or turn off PWM outputs without disabling the PWM modulator.

Some motor drive control schemes require asymmetrical center-based PWM waveforms, shown in Figure 11. This requires different modulation levels in the first phase and second phase of the PWM cycle. Depending on the control algorithm, the user can update both modulation registers together at the beginning of the PWM cycle or update each register before the beginning of each PWM timer phase.

In symmetrical PWM mode, the user must only update the first modulation register. Control bits in PWM_xCTL registers enable the selection of symmetrical or asymmetrical modulation modes for each phase. The DUEN control bit in the PWM_CTL register determines if the modulation registers take effect once or twice per PWM cycle.

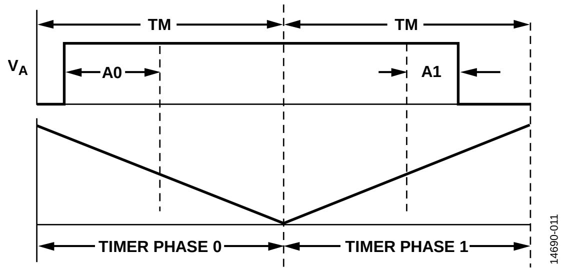

Figure 11. Asymmetrical PWM Waveforms.

The PWM controller includes a function that chops the PWM waveform at high frequency to support transformer-coupled gate drivers. Bit fields of the PWM_CHANCFG register enables this feature.

The resolution of the duty-cycle register is determined by the system clock and the PWM frequency ($f_{SYSCLK}/(2 \times f_{PWM})$). For example, if f_{SYSCLK} = 80 MHz and f_{PWM} = 10 kHz, the duty cycle gets quantized into 4000 step or ~12 bits. At lower PWM frequencies, the resolution is sufficiently high, but as the PWM frequency increases, the resolution can fall short in some applications. For these, the PWM controller offers a heightened precision pulse output mode that increases the resolution four times (2 bits). For more information, refer to the ADSP-CM40x Mixed-Signal Control Processor with ARM Cortex-M4 Hardware Reference Manual.

Output Control and Trip Handling

The output controller manages the hardware interface between the PWM modulator and the power inverter. Level shifting or fully isolating gate drive circuits buffer the logic level IC outputs from the power transistor gates. Control bits in the PWM_CHANCFG register select the active polarity for each PWM output pin to match the behavior of the gate drivers.

Overload current detection circuits protect the power inverter hardware by providing a PWM trip signal to shut off the PWM outputs in the event of a fault. The PORT unit routes the TRIP0 PWM fault signal asynchronously from the TRIP0 input pin to ensure safe shutdown of the PWM outputs even in the event of a system clock failure.

The TRU synchronously routes internal fault signals from internal units such as the SINC filter to the TRIP1 input. Figure 12 describes an example of connecting two overload trigger signals from the internal SINC filter and an external hardware trip signal to the TRIP0 and TRIP1 inputs.

Control bits in the PWM_TRIPCFG register select trip sources and trip mode for each PWM channel. AC motor controllers normally require the fault trip mode, which shuts down the PWM modulator until the CPU clears the fault. Status bits in the PWM_STAT register indicate the PWM channels that have tripped and the CPU must write one to these bit locations to clear the fault and restart the PWM modulator.

The self restart trip mode is useful for peak current mode control sometimes used in digital power control.

The exact restart sequence after an inverter trip event depends on the end application requirements. The safest approach is to wait until the motor has come to a complete stop before allowing a controlled motor start. A spinning motor, even an induction motor, can generate winding back EMFs and the inverter must match these voltages before reconnecting to avoid significant overload currents.

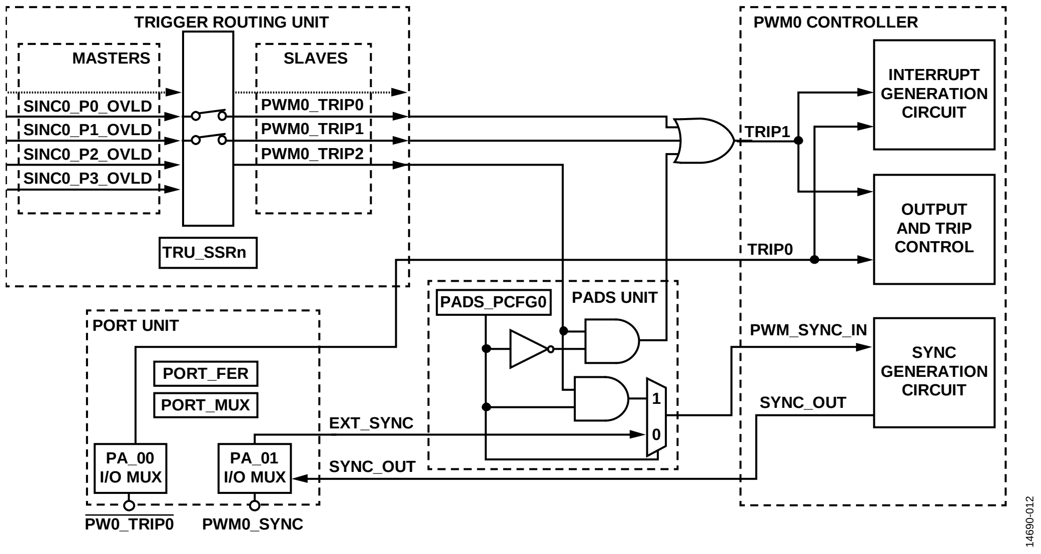

Figure 12. Routing and Pin Multiplexing for SYNC and TRIP Signals

Interrupt Generation and Trigger Routing

The PWM modulator is the timing engine for the motor control algorithm. The interrupt controller and trigger routing units support synchronization of the modulator with signal sampling and algorithm execution. These units also handle fault signal routing and notification. The CPU must acknowledge trigger and interrupt signals by clearing the appropriate flags to ensure reliable operation of the PWM controller.

The interrupt controller generates a single PWM_SYNC interrupt trigger from any of the five different timer sources (TIMER0 to TIMER4) on the period boundaries. The PWM_IMSK register includes the control bits to unmask one or more of these interrupts and enable the PWM_SYNC interrupt. The PWM_ILAT register includes corresponding latched status bits to indicate the interrupt event.

The PWM_SYNC interrupt service routine (ISR) code must write 1 to the PWM_ILAT register status bit to clear the interrupt and allow the generation of the next interrupt. The user can use the ISR to call the control routine that updates the modulation duty cycle registers for the next PWM cycle but it is more likely the user uses an ADC interrupt to call the control routine to update the duty cycle registers based on motor feedback signals.

For the motor control algorithm to run with optimum performance, it is required to synchronize the ADC, SINC, and other peripherals to PWM_SYNC signal. PWM_SYNC can trigger an interrupt but PWM_SYNC can also act as a trigger master signal and can be routed through the TRU to the desired peripheral slave by loading the master trigger identification into the appropriate TRU slave trigger register (TRU_SSRn).

The interrupt generating PWM_SYNC signal and the trigger master PWM_SYNC are the same signal. Using either signal requires the PWM_SYNC interrupt to be serviced on each event for continuous SYNC trigger generation.

The most common source for the SYNC master is the PWM timer itself. Typically, PWM_SYNC is routed through the TRU to the embedded ADC, SINC and other internal peripherals. To synchronize external devices, such as an ADC or a field-programmable gate array (FPGA), PWM_SYNC can be routed to a port pin as PWM_SYNC_OUT.

If the controller is part of a multiaxis system, the PWM_SYNC signal usually comes from a system master. In this case, the master signal connects to the external PWM_SYNC pin that resets the timer as described earlier. In this operating mode, the PWM timer still generates an internal SYNC signal. Think of the external SYNC as a hard reset of the counter.

The ADSP-CM402F/ADSP-CM403F/ADSP-CM407F/ADSP-CM408F/ADSP-CM409F has three PWM blocks and is capable of controlling multiple motors at the same time. If the system has an external master, use the SYNC pin as described previously. However, in multiaxis standalone applications, there is no master to provide the required synchronizing pulse, so the IC must manage synchronization. To handle this, the ADSP-CM402F/ADSP-CM403F/ADSP-CM407F/ADSP-CM408F/ADSP-CM409F offers a Trigger Slave mode. As shown in Figure 12, Trigger Slave mode repurposes the PWM_TRIP_TRIG2 signal and the PWM_SYNC_IN signal, enabling the synchronization of multiple PWM blocks without using external pins. The alterative use of the PWM_TRIP_TRIG2 signal and the PWM_SYNC_IN signal is enabled through the PADS unit by setting the PADC_PFC0.PWM0_SYNC_TRU = 1 (default = 0). In this mode, do not use TRIP2 as a fault trip signal.

A typical use of the Trigger Slave mode synchronizes PWM blocks by connecting all PWM_TRIP_TRIG2 signals to a general-purpose timer trigger master through the TRU.

The user must enable PWM trip interrupts to allow the CPU to respond to any PWM trip events and clear the interrupt before restarting the motor in a controlled way. There is a single PWM trip interrupt vector for each PWM controller. The PWM_IMSK and PWM_ILAT registers includes the control bits to unmask and clear interrupts for the PWM_TRIP0 signal and PWM_TRIP1 signal. Typically, the PWM trip ISR records the fault, clears the interrupt, and calls the fault handing routine defined for the application.

Pin Multiplexer Configuration

The PORT unit is a pin multiplexer that connects signals between the microcontroller peripherals and the digital input/output pins. The PORT control registers set the input/output behavior and peripheral connection of individual input/output pins according to the defined application requirements. There are six input/output pin groups with up to 16 pins in each group. Each pin can have up to four peripheral signal connections. Pins can operate in either peripheral mode or general-purpose input/output (GPIO) mode. The ADSP-CM402F/ADSP-CM403F/ADSP-CM407F/ADSP-CM408F/ADSP-CM409F data sheet defines the available peripheral connections for each pin. The required PORT configuration settings depend on the hardware design, which defines the pins connected to the gate drive and overload connection circuits.

The PORT control registers that support peripheral connections are the PORTx_FER registers and the PORTx_MUX registers, where x = A, B, C, D, E, or F. Setting bits in the PORTx_FER registers define pins as peripheral output functions rather than GPIO and setting bits in the PORTx_MUX registers define the peripheral signals connected to the pins.

PWM Output Control

The inverter voltage function is the last step in a sequence of calculations to control the motor. The control algorithm calculates the modulation values using Equation 4, which defines the duty cycle as a function of the target voltage, the dc bus voltage, and the PWM period. The channel duty cycle registers, PWM_AH, PWM_BH, and PWM_CH, are double-buffered registers so new values only take effect on period boundaries. The only requirement is that the register update occurs in advance of the next period boundary. The PWM_SYNC trigger provides the timing signal for the interrupt service routine that calls the control algorithm and updates the duty cycle registers.

PWM CONTROL PROGRAMMING EXAMPLE

Open loop V/Hz control is a good example to illustrate the use of the PWM controller in 3-phase ac motor applications. The user defines the target speed by setting the ac motor frequency. The V/Hz control law requires the ac voltage magnitude be a linear function of motor frequency. A sinusoidal modulation function integrates the angular frequency to calculate the electrical angle and the three sine functions offset by 120° electrical. It then scales the output of the sine function by the voltage magnitude. The digital control version increments the electrical angle and calculates new inverter voltage values on each time step (T_S).

V/Hz Program Flow

Figure 13 describes the V/Hz control algorithm and Figure 14 describes the complete program flow. The additional functions include system initialization, user command interface, start/stop sequencing, and fault handling. The main program calls the initialization routines while interrupt service routines call all the other functions. This system example uses the PWM0 controller with external and internal PWM trips.

When using the maximum system clock frequency of 100 MHz, a typical PWM frequency of 10 kHz and 2 µs dead time, the period register value is 5000 and the dead time register value is 100. The output voltage resolution in this case is approximately 12 bits.

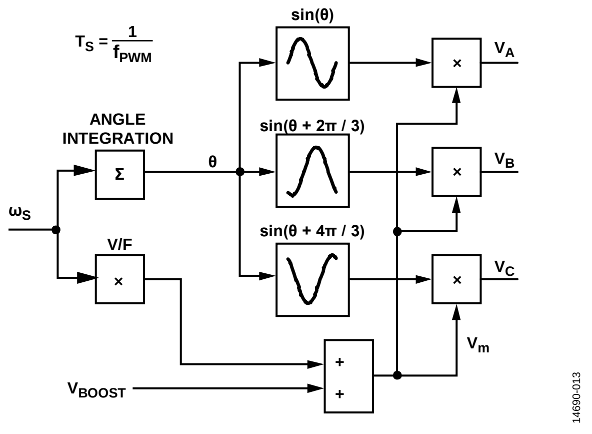

Figure 13. V/Hz Control Algorithm

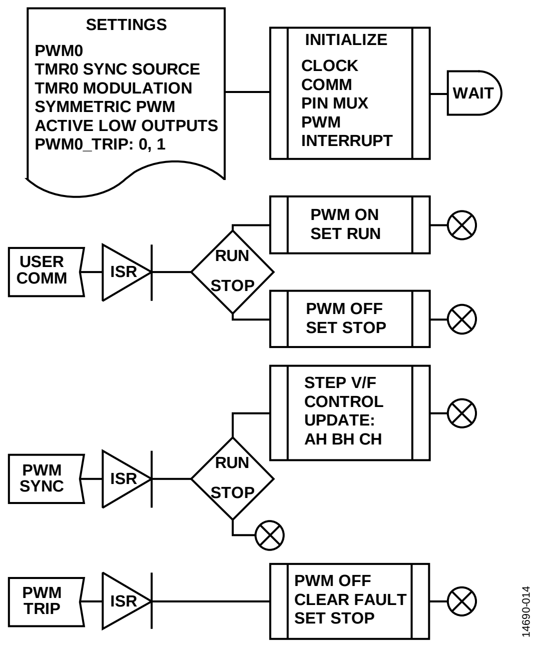

Figure 14. V/Hz Program Flow

CODE EXAMPLE

Visit https://ez.analog.com/docs/DOC-12643 for a code example that shows how to set up PWM on the ADSP-CM402F/ADSP-CM403F/ADSP-CM407F/ADSP-CM408F/ADSP-CM409F. The example configures PWM0 for a 3-phase inverter with three high and three low side PWM signals. PWM frequency is 10 kHz and dead time is 1 µs. The code also shows how to service the PWM_SYNC interrupt and ITRIP fault handling.

APPENDIX

DOUBLE-BUFFERED CONTROL REGISTERS

Table 1 shows the list of double-buffered control registers. The user can write to these registers at any time during the PWM cycle. However, the writes do not take effect until the next period boundary (PWM_SYNC).

Table 1. Double-Buffered Control Registers

Register Name	Suffix	Function	Example
PWM_TMx	x = 0, 1, 2 ,3, 4	Timer period	PWM0_TM0
PWM_DLYx	x = 0, 1, 2, 3, 4	Timer delay	PWM0_DLY0
PWM_DT	Not applicable	Dead time	PWM0_DT
PWM_xCTL	x =A, B, C, D	Channel control	PWM0_ACTL

PWM_xHn	x =A, B, C, D
	n = 0, 1
		High channel duty cycle (integer)	PWM0_AH0

PWM_xLn	x = A, B, C, D
	n = 0, 1
		Low channel duty cycle (integer)	PWM0_AL0

PWM_xYn_HP	x =A, B, C, D
	Y = H, L
	n = 0, 1
		Heightened precision duty cycle	PWM0_AH0_HP

PWM_xY_dutyn	x =A, B, C, D
	Y = H, L
	n = 0,1
		Full duty cycle (Q15.8 format)	PWM0_AH_duty0

REGISTER SETTINGS

Table 2 to Table 6 highlight the important register settings used in 3-phase ac motor control. The third column lists enumeration settings defined in the register definitions, included in the defCM40z.h file supplied with the ADSP-CM40x Enablement Package.

Table 2. PWM Control Register Settings PWM_CTL

Bit Field Name	Bit Value	ENUM_PWM_CTL
INTSYNCREF[1]	0	INTSYNC_0
EXTSYNCSEL[2]	1	EXTSYNC_SYNC
EXTSYNC[1]	0	INTSYNC
ADEN[2]	0	Set for asynchronous dead time
DLYDEN[2]	0	Timer delay not used
DLYCEN[2]	0	Timer delay not used
DLYBEN[2]	0	Timer delay not used
DLYAEN[2]	0	Timer delay not used
DUEN[2]	0	Set for double-update mode
SWTRIP[2]	0	Set to force a PWM trip
EMRUN[1]	0	EMURUN_DIS
GLOBEN[1]	x	PWM_EN or PWM_DIS

[1] Register needed to configure the PWM timer for a standard 3-phase application.
[2] Register configures advanced options.

Table 3. Channel Configuration Register Settings (PWM_CHANCFG)

Bit Field Name	Value	ENUM_PWM_CHANCFG
ENCHOPDL[2]	0	Channel D not used
POLDL[2]	0	Channel D not used
ENHPDH[2]	0	Channel D not used
ENCHOPDH[2]	0	Channel D not used
POLDH[2]	0	Channel D not used
MODELSD[2]	0	Channel D not used
REFTMRD[2]	0	Channel D not used
ENCHOPCL[2]	0	Gate chopping not used
POLCL[1]	0	CL_ACTLO
ENHPCH[2]	0	Heightened precision PWM not used
ENCHOPCH[2]	0	Gate chopping not used
POLCH[1]	0	CH_ACTLO
MODELSC[1]	0	LOC_INVHI
REFTMRC[1]	0	REFTMRC_0
ENCHOPBL[2]	0	Gate chopping not used
POLBL[1]	0	BL_ACTLO
ENHPBH[2]	0	Heightened precision PWM not used
ENCHOPBH[2]	0	Gate chopping not used
POLBH[1]	0	BH_ACTLO
MODELSB[1]	0	LOB_INVHI
REFTMRB[1]	0	REFTMRB_0
ENCHOPAL[2]	0	Gate chopping not used
POLAL[1]	0	AL_ACTLO
ENHPAH[2]	0	Heightened precision PWM not used
ENCHOPAH[2]	0	Gate chopping not used
POLAH[1]	0	BH_ACTLO
MODELSA[1]	0	LOB_INVHI
REFTMRA[1]	0	REFTMRB_0

[1] Register needed to configure the PWM timer for a standard 3-phase application.
[2] Register configures advanced options.

Table 4. Channel Control Register Settings (PWM_xCTL)

Bit Field Name	Value	ENUM_PWM_CTL
PULSEMODELO[1]	0	SYM_LO
PULSEMODEHI[1]	0	SYM_HI
XOVR[2]	0	XOVR_DIS
DISLO[1]	1	LO_EN
DISHI[1]	1	HI_EN

[1] Register needed to configure the PWM timer for a standard 3-phase application.
[2] Register configures advanced options.

Table 5. Trip Configuration Register Settings (PWM_TRIPCFG)

Bit Field Name	Value	
MODE1D	0	Channel D not used
EN1D	0	Channel D not used
MODE0D	0	Channel D not used
EN0D	0	Channel D not used
MODE1C	0	TRIP1C_FLT
EN1C	0	TRIP1C_EN
MODE0C	0	TRIP0C_FLT
EN0C	0	TRIP0C_EN
MODE1B	0	TRIP1B_FLT
EN1B	0	TRIP1B_EN
MODE0B	0	TRIP0B_FLT
EN0B	0	TRIP0B_EN
MODE1A	0	TRIP1A_FLT
EN1A	0	TRIP1A_EN
MODE0A	0	TRIP0A_FLT
EN0A	0	TRIP0A_EN

Table 6. Interrupt Mask Register Settings (PWM_IMSK)

Bit Field Name	Value	ENUM_PWM_IMSK
TMR4PER	0	PWMTMR4 not used
TMR3PER	0	PWMTMR4 not used
TMR2PER	0	PWMTMR4 not used
TMR1PER	0	PWMTMR4 not used
TMR0PER[1]	1	PER0_UMSK
TRIP1[1]	1	TRIP1_UMSK
TRIP0[1]	1	TRIP0_UMSK

[1] Register needed to configure the PWM timer for a standard 3-phase application.