

Pulse-Width Modulation Subsystem (PWMSS)

This chapter describes the PWMSS of the device.

Topic	Page
15.1 Pulse-Width Modulation Subsystem (PWMSS).....	2321
15.2 Enhanced PWM (ePWM) Module	2330
15.3 Enhanced Capture (eCAP) Module	2465
15.4 Enhanced Quadrature Encoder Pulse (eQEP) Module	2507

15.1 Pulse-Width Modulation Subsystem (PWMSS)

15.1.1 Introduction

15.1.1.1 Features

The general features of the PWMSS are:

eHRPWM

- Dedicated 16 bit time-base with Period / Frequency control
- Can support 2 independent PWM outputs with Single edge operation
- Can support 2 independent PWM outputs with Dual edge symmetric operation
- Can support 1 independent PWM output with Dual edge asymmetric operation
- Supports Dead-band generation with independent Rising and Falling edge delay control
- Provides asynchronous over-ride control of PWM signals during fault conditions
- Supports “trip zone” allocation of both latched and un-latched fault conditions
- Allows events to trigger both CPU interrupts and start of ADC conversions
- Support PWM chopping by high frequency carrier signal, used for pulse transformer gate drives.
- High-resolution module with programmable delay line.
 - Programmable on a per PWM period basis.
 - Can be inserted either on the rising edge or falling edge of the PWM pulse or both or not at all.

eCAP

- Dedicated input Capture pin
- 32 bit Time Base (counter)
- 4 x 32 bit Time-stamp Capture registers (CAP1-CAP4)
- 4 stage sequencer (Mod4 counter) which is synchronized to external events (ECAPx pin edges)
- Independent Edge polarity (Rising / Falling edge) selection for all 4 events
- Input Capture signal pre-scaling (from 1 to 16)
- One-shot compare register (2 bits) to freeze captures after 1 to 4 Time-stamp events
- Control for continuous Time-stamp captures using a 4 deep circular buffer (CAP1-CAP4) scheme
- Interrupt capabilities on any of the 4 capture events

eQEP

- Input Synchronization
- Quadrature Decoder Unit
- Position Counter and Control unit for position measurement
- Quadrature Edge Capture unit for low speed measurement
- Unit Time base for speed/frequency measurement
- Watchdog Timer for detecting stalls

15.1.1.2 Unsupported Features

The PWMSS module features not supported are shown in [Table 15-1](#).

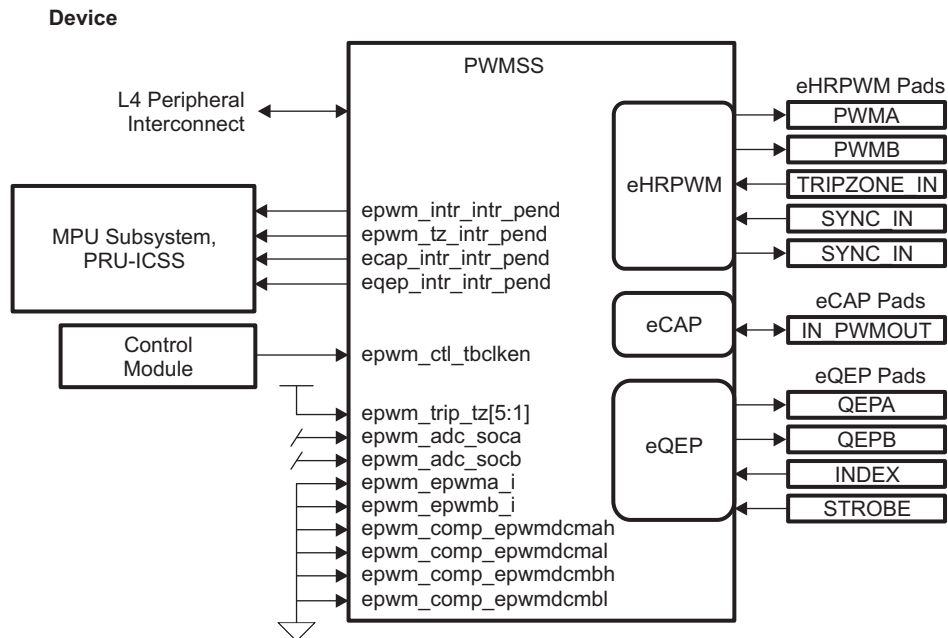
Table 15-1. Unsupported Features

Feature	Reason
ePWM inputs	Not pinned out
ePWM tripzone 1-5 inputs	Only Tripzone0 is pinned out
ePWM digital comparators	Inputs not connected
eQEP quadrature outputs	Only input signals are connected
eCAP3-5	Module not used
eQEP3-5	Module not used

15.1.2 Integration

The Pulse Width Modulation Subsystem (PWMSS) includes a single instance of the Enhanced High Resolution Pulse Width Modulator (eHRPWM), Enhanced Capture (eCAP), and Enhanced Quadrature Encoded Pulse (eQEP) modules. This device includes three instantiations of the PWMSS.

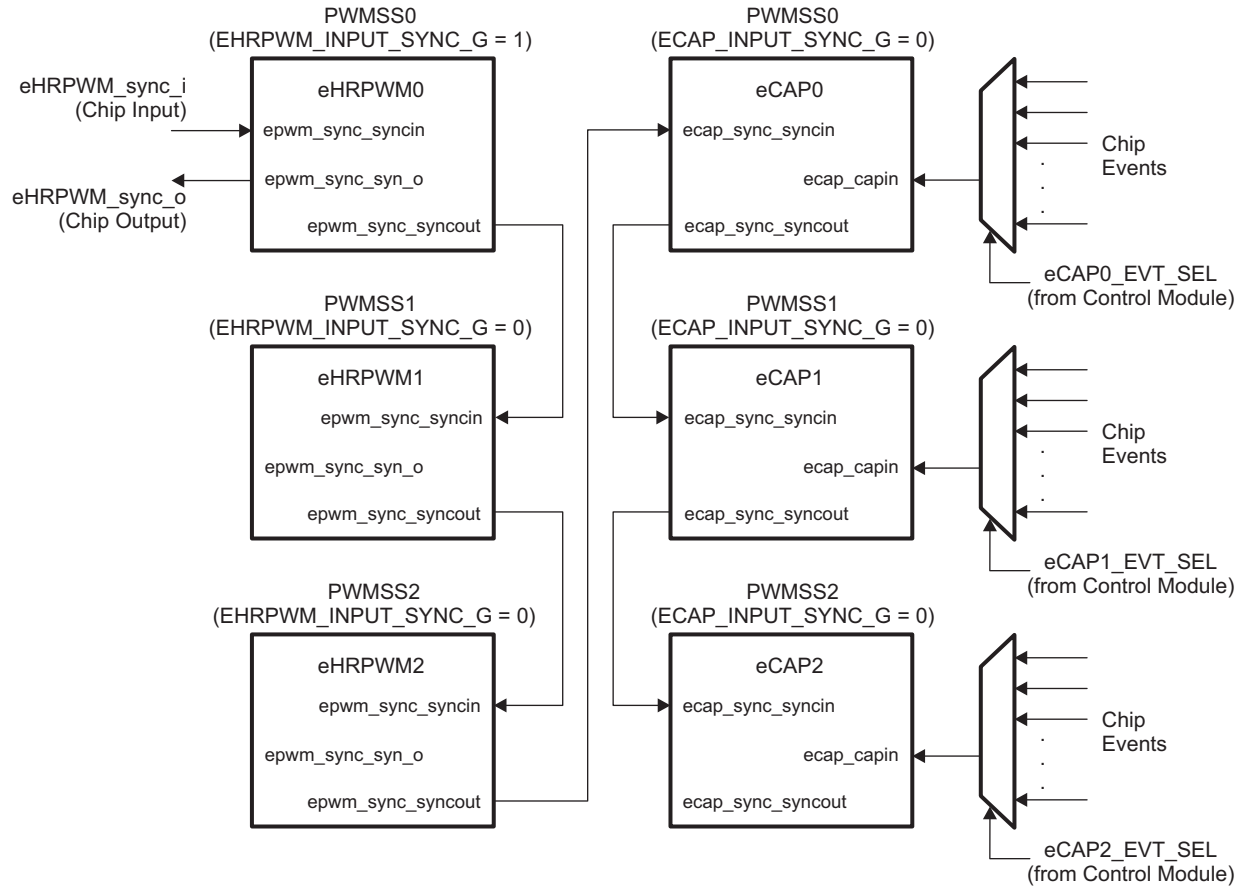
Figure 15-1. PWMSS Integration



15.1.2.1 PWMSS Synchronization Detail

The PWM (eHRPWM) and capture (eCAP) components of the PWMSS provide synchronization signals to allow them to be synchronized to other modules or events. On this device, these signals are connected in a daisy-chain fashion as shown in Figure 15-2.

The eCAP capture events may be selected from among 31 different pins or internal interrupt signals. The event is selected using the corresponding ECAPx_EVTCAPT field of the ECAP_EVT_CAP register in the Control Module.

Figure 15-2. PWMSS Synchronization


15.1.2.2 PWMSS Connectivity Attributes

The general connectivity attributes for the PWMSS module are shown in [Table 15-2](#).

Table 15-2. PWMSS Connectivity Attributes

Attributes	Type
Power Domain	Peripheral Domain
Clock Domain	PD_PER_L4LS_GCLK
Reset Signals	PER_DOM_RST_N
Idle/Wakeup Signals	Smart Idle
Interrupt Requests	2 ePWM interrupts per instance epwm_intr_intr - Event interrupt, ePWMxINT for ARM subsystem, epwm_intr_intr_pend for PRU-ICSS epwm_tz_intr - Tripzone interrupt, ePWMx_TZINT for ARM subsystem, pwm_trip_zone for PRU-ICSS (only 1 for all 3 instances) 1 eCAP interrupt per instance ecap_intr - Capture/PWM event interrupt, eCAPxINT for ARM subsystem, ecap_intr_intr_pend for PRU-ICSS 1 eQEP Interrupt per instance eqep_intr_intr - Event interrupt, eQEPxINT for ARM subsystem, eqep_intr_intr_pend for PRU-ICSS (only for eQEP0)

Table 15-2. PWMSS Connectivity Attributes (continued)

Attributes	Type
DMA Requests	Interrupt requests are redirected as DMA requests: <ul style="list-style-type: none"> • 1 DMA request from ePWM per instance (ePWMEVTx) • 1 DMA request from eCAP per instance (eCAPEVTx) • 1 DMA request from eQEP per instance (eQEPEVTx)
Physical Address	L4 Peripheral slave port

15.1.2.3 PWMSS Clock and Reset Management

The PWMSS controllers have separate bus interface and functional clocks.

Table 15-3. PWMSS Clock Signals

Clock Signal	Max Freq	Reference / Source	Comments
PWMSS_ocp_clk Interface / Functional clock	100 MHz	CORE_CLKOUTM4 / 2	pd_per_l4ls_gclk from PRCM

15.1.2.4 PWMSS Pin list

The external signals for the PWMSS module are shown in the following table.

Table 15-4. PWMSS Pin List

Pin	Type	Description
EPWMxA	O	PWM output A
EPWMxB	O	PWM output B
EPWM_SYNCIN	I	PWM Sync input
EPWM_SYNCOUT	O	PWM Sync output
EPWM_TRIPZONE[5:0]	I	PWM Tripzone inputs
ECAP_CAPIN_APWMOUT	I/O	eCAP Capture input / PWM output
EQEP_A	I/O	eQEP Quadrature input/output
EQEP_B	I/O	eQEP Quadrature input/output
EQEP_INDEX	I/O	eQEP Index input/output
EQEP_STROBE	I/O	eQEP Strobe input/output

15.1.3 PWMSS Registers

Table 15-5 lists the memory-mapped registers for the PWMSS. All register offset addresses not listed in Table 15-5 should be considered as reserved locations and the register contents should not be modified.

Table 15-5. PWMSS Registers

Offset	Acronym	Register Name	Section
0h	IDVER	IP Revision Register	Section 15.1.3.1
4h	SYSCONFIG	System Configuration Register	Section 15.1.3.2
8h	CLKCONFIG	Clock Configuration Register	Section 15.1.3.3
Ch	CLKSTATUS	Clock Status Register	Section 15.1.3.4

15.1.3.1 IDVER Register (offset = 0h) [reset = 4000000h]

IDVER is shown in [Figure 15-3](#) and described in [Table 15-6](#).

The IP revision register is used by software to track features, bugs, and compatibility.

Figure 15-3. IDVER Register

31	30	29	28	27	26	25	24
SCHEME		RESERVED			FUNC		
R-1h		R-0h			R-0h		
23	22	21	20	19	18	17	16
FUNC							
R-0h							
15	14	13	12	11	10	9	8
R_RTL				X_MAJOR			
R-0h				R-0h			
7	6	5	4	3	2	1	0
CUSTOM		Y_MINOR					
R-0h		R-0h					

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-6. IDVER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SCHEME	R	1h	Used to distinguish between the old scheme and current.
29-28	RESERVED	R	0h	
27-16	FUNC	R	0h	FUNC
15-11	R_RTL	R	0h	RTL version (R), maintained by IP design owner.
10-8	X_MAJOR	R	0h	Major revision (X)
7-6	CUSTOM	R	0h	CUSTOM
5-0	Y_MINOR	R	0h	Minor revision (Y)

15.1.3.2 SYSCONFIG Register (offset = 4h) [reset = 28h]

 SYSCONFIG is shown in [Figure 15-4](#) and described in [Table 15-7](#).

The system configuration register is used for clock management configuration.

Figure 15-4. SYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		STANDBYMODE		IDLEMODE		FREEEMU	SOFTRESET
R-0h		R/W-2h		R/W-2h		R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-7. SYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	
5-4	STANDBYMODE	R/W	2h	Configuration of the local initiator state management mode. By definition, initiator may generate read/write transaction as long as it is out of STANDBY state. 0h = Force-standby mode: local initiator is unconditionally placed in standby state. Backup mode, for debug only. 1h = No-standby mode: local initiator is unconditionally placed out of standby state. Backup mode, for debug only. 2h = Smart-standby mode: local initiator standby status depends on local conditions, i.e., the module's functional requirement from the initiator. IP module should not generate (initiator-related) wakeup events. 3h = Reserved.
3-2	IDLEMODE	R/W	2h	Configuration of the local target state management mode. By definition, target can handle read/write transaction as long as it is out of IDLE state. 0h = Force-idle mode: local target's idle state follows (acknowledges) the system's idle requests unconditionally, i.e. regardless of the IP module's internal requirements. Backup mode, for debug only. 1h = No-idle mode: local target never enters idle state. Backup mode, for debug only. 2h = Smart-idle mode: local target's idle state eventually follows (acknowledges) the system's idle requests, depending on the IP module's internal requirements. IP module shall not generate (IRQ- or DMA-request-related) wakeup events. 3h = Reserved.
1	FREEEMU	R/W	0h	Sensitivity to emulation (debug) suspend event from Debug Subsystem. 0h = IP module is sensitive to emulation suspend. 1h = IP module is not sensitive to emulation suspend event. Debug suspend event is ignored.
0	SOFTRESET	R/W	0h	Software reset (optional)

15.1.3.3 CLKCONFIG Register (offset = 8h) [reset = 111h]

CLKCONFIG is shown in [Figure 15-5](#) and described in [Table 15-8](#).

The clock configuration register is used in the PWMSS submodule for clkstop req and clk_en control.

Figure 15-5. CLKCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						ePWMCLKSTO P_REQ	ePWMCLK_EN
R-0h						R/W-0h	R/W-1h
7	6	5	4	3	2	1	0
RESERVED		eQEPCLKSTO P_REQ	eQEPCLK_EN	RESERVED		eCAPCLKSTO P_REQ	eCAPCLK_EN
R-0h		R/W-0h	R/W-1h	R-0h		R/W-0h	R/W-1h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-8. CLKCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
9	ePWMCLKSTOP_REQ	R/W	0h	This bit controls the clkstop_req input to the ePWM module.
8	ePWMCLK_EN	R/W	1h	This bit controls the clk_en input to the ePWM module.
7-6	RESERVED	R	0h	
5	eQEPCLKSTOP_REQ	R/W	0h	This bit controls the clkstop_req input to the eQEP module
4	eQEPCLK_EN	R/W	1h	This bit controls the clk_en input to the eQEP module.
3-2	RESERVED	R	0h	
1	eCAPCLKSTOP_REQ	R/W	0h	This bit controls the clkstop_req input to the eCAP module.
0	eCAPCLK_EN	R/W	1h	This bit controls the clk_en input to the eCAP module.

15.1.3.4 CLKSTATUS Register (offset = Ch) [reset = 0h]

CLKSTATUS is shown in [Figure 15-6](#) and described in [Table 15-9](#).

The clock status register is used in the PWMSS submodule for clkstop ack and clk_en ack status.

Figure 15-6. CLKSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						ePWM_CLKST OP_ACK	ePWM_CLK_E N_ACK
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED		eQEP_CLKST OP_ACK	eQEP_CLK_EN _ACK	RESERVED		eCAP_CLKST OP_ACK	eCAP_CLK_EN _ACK
R-0h		R-0h	R-0h	R-0h		R-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-9. CLKSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
9	ePWM_CLKSTOP_ACK	R	0h	This bit is the clkstop_req_ack status output of the ePWM module.
8	ePWM_CLK_EN_ACK	R	0h	This bit is the clk_en status output of the ePWM module.
7-6	RESERVED	R	0h	
5	eQEP_CLKSTOP_ACK	R	0h	This bit is the clkstop_req_ack status output of the eQEP module.
4	eQEP_CLK_EN_ACK	R	0h	This bit is the clk_en status output of the eQEP module.
3-2	RESERVED	R	0h	
1	eCAP_CLKSTOP_ACK	R	0h	This bit is the clkstop_req_ack status output of the eCAP module.
0	eCAP_CLK_EN_ACK	R	0h	This bit is the clk_en status output of the eCAP module.

15.2 Enhanced PWM (ePWM) Module

15.2.1 Introduction

An effective PWM peripheral must be able to generate complex pulse width waveforms with minimal CPU overhead or intervention. It needs to be highly programmable and very flexible while being easy to understand and use. The ePWM unit described here addresses these requirements by allocating all needed timing and control resources on a per PWM channel basis. Cross coupling or sharing of resources has been avoided; instead, the ePWM is built up from smaller single channel modules with separate resources and that can operate together as required to form a system. This modular approach results in an orthogonal architecture and provides a more transparent view of the peripheral structure, helping users to understand its operation quickly.

In this chapter, the letter x within a signal or module name is used to indicate a generic ePWM instance on a device. For example, output signals EPWMxA and EPWMxB refer to the output signals from the ePWMx instance. Thus, EPWM1A and EPWM1B belong to ePWM1 and, likewise, EPWM4A and EPWM4B belong to ePWM4.

15.2.1.1 Submodule Overview

The ePWM module represents one complete PWM channel composed of two PWM outputs: EPWMxA and EPWMxB. Multiple ePWM modules are instanced within a device as shown in [Figure 15-7](#). Each ePWM instance is identical with one exception. Some instances include a hardware extension that allows more precise control of the PWM outputs. This extension is the high-resolution pulse width modulator (HRPWM) and is described in [Section 15.2.2.10](#). See [Section 15.1.2](#) to determine which ePWM instances include this feature. Each ePWM module is indicated by a numerical value starting with 0. For example ePWM0 is the first instance and ePWM2 is the third instance in the system and ePWMx indicates any instance.

The ePWM modules are chained together via a clock synchronization scheme that allows them to operate as a single system when required. Additionally, this synchronization scheme can be extended to the capture peripheral modules (eCAP). The number of modules is device-dependent and based on target application needs. Modules can also operate stand-alone.

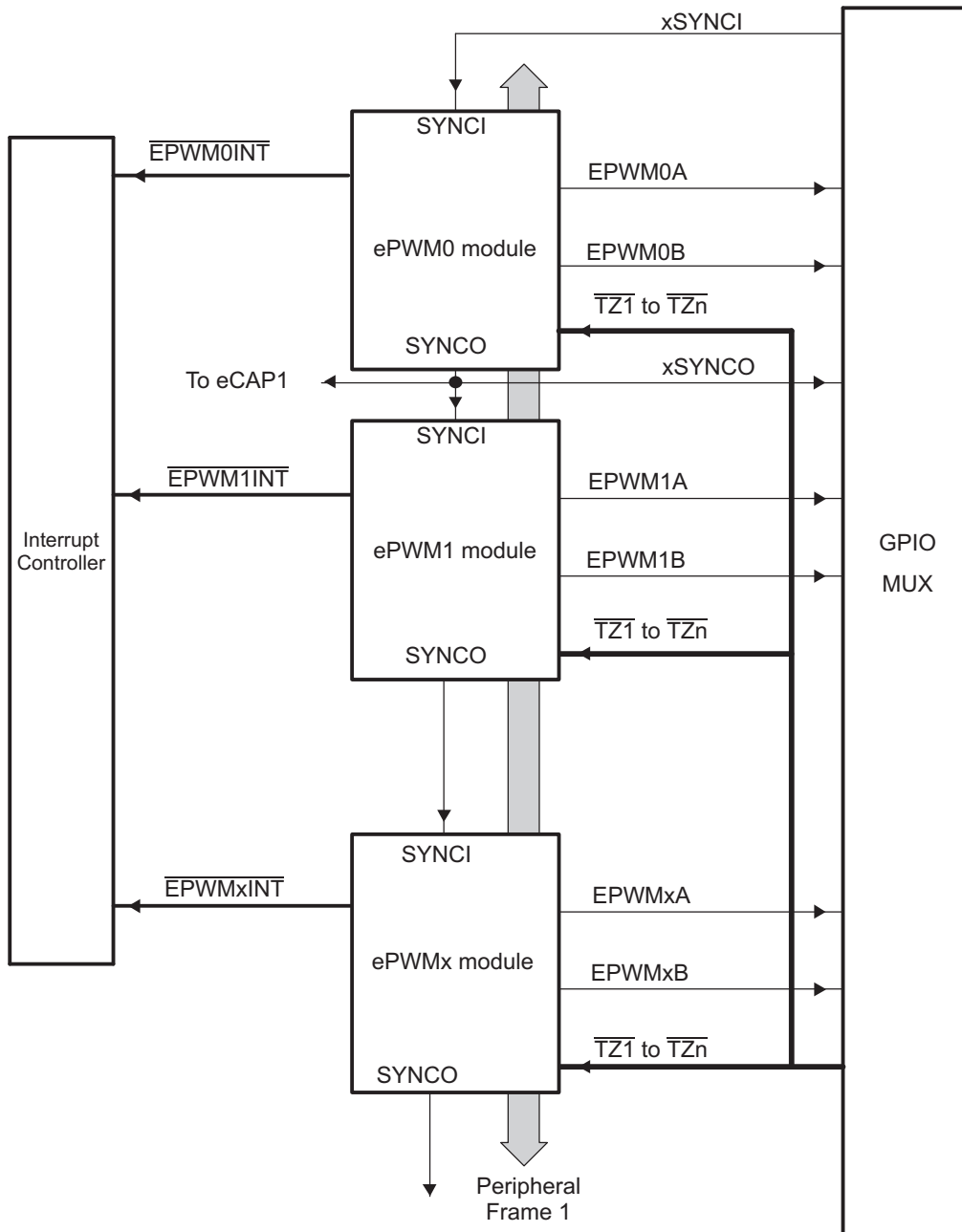
Each ePWM module supports the following features:

- Dedicated 16-bit time-base counter with period and frequency control
- Two PWM outputs (EPWMxA and EPWMxB) that can be used in the following configurations::
 - Two independent PWM outputs with single-edge operation
 - Two independent PWM outputs with dual-edge symmetric operation
 - One independent PWM output with dual-edge asymmetric operation
- Asynchronous override control of PWM signals through software.
- Programmable phase-control support for lag or lead operation relative to other ePWM modules.
- Hardware-locked (synchronized) phase relationship on a cycle-by-cycle basis.
- Dead-band generation with independent rising and falling edge delay control.
- Programmable trip zone allocation of both cycle-by-cycle trip and one-shot trip on fault conditions.
- A trip condition can force either high, low, or high-impedance state logic levels at PWM outputs.
- Programmable event prescaling minimizes CPU overhead on interrupts.
- PWM chopping by high-frequency carrier signal, useful for pulse transformer gate drives.

Each ePWM module is connected to the input/output signals shown in [Figure 15-7](#). The signals are described in detail in subsequent sections.

The order in which the ePWM modules are connected may differ from what is shown in [Figure 15-7](#). See [Section 15.2.2.3.3.2](#) for the synchronization scheme for a particular device. Each ePWM module consists of seven submodules and is connected within a system via the signals shown in [Figure 15-8](#).

Figure 15-7. Multiple ePWM Modules



NOTE: Figure 15-7 is a generic block diagram. For specific implementation, see Section 15.1.2.1, PWMSS Synchronization Detail.

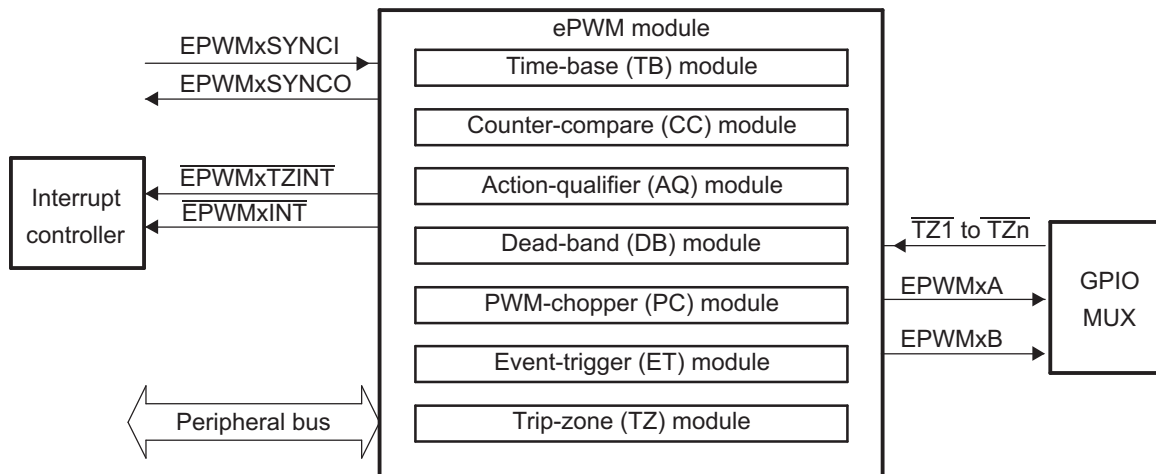
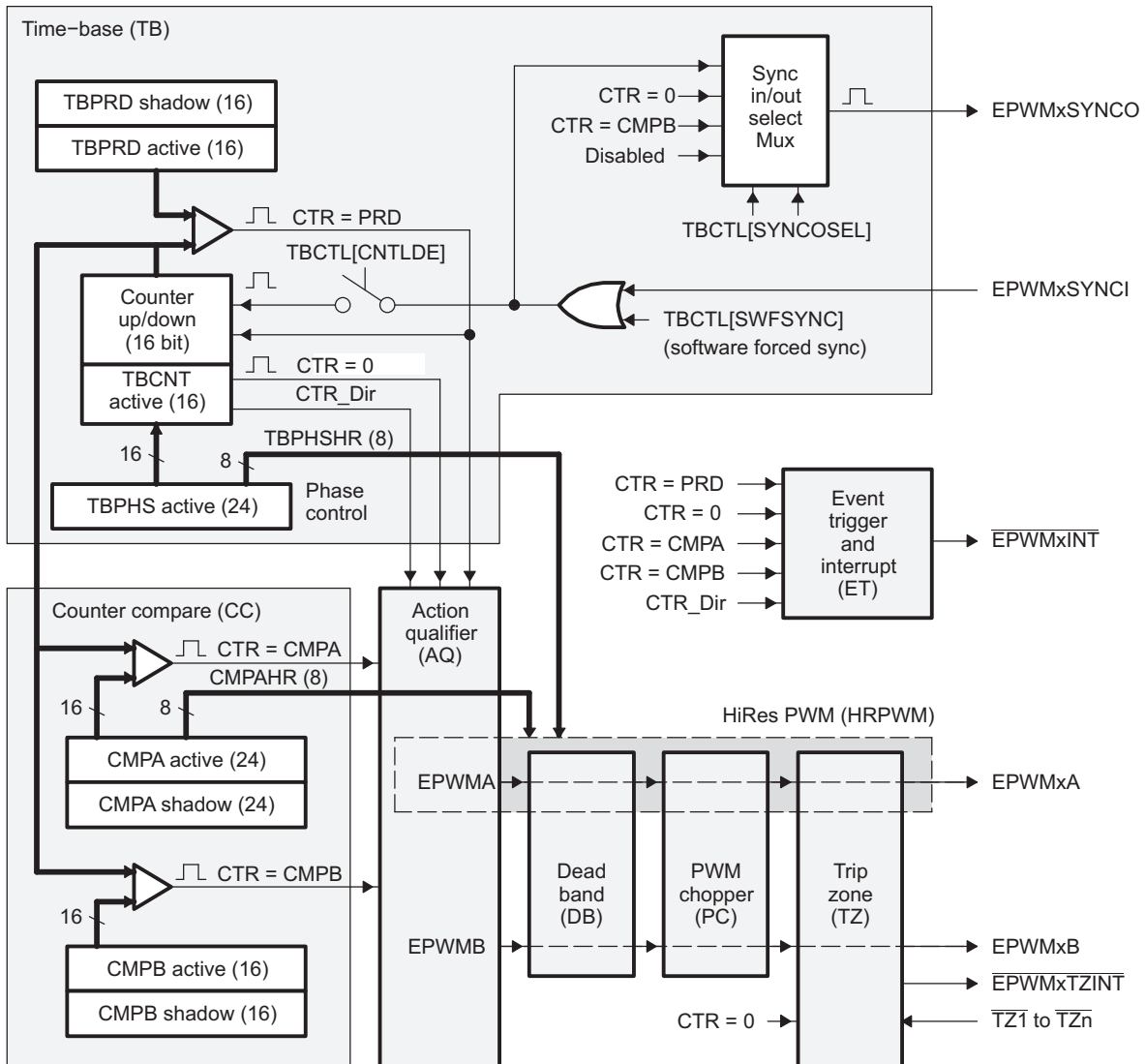
Figure 15-8. Submodules and Signal Connections for an ePWM Module


Figure 15-9 shows more internal details of a single ePWM module. The main signals used by the ePWM module are:

- **PWM output signals (EPWMxA and EPWMxB).** The PWM output signals are made available external to the device through the GPIO peripheral described in the system control and interrupts guide for your device.
- **Trip-zone signals ($\overline{\text{TZ1}}$ to $\overline{\text{TZn}}$).** These input signals alert the ePWM module of an external fault condition. Each module on a device can be configured to either use or ignore any of the trip-zone signals. The trip-zone signal can be configured as an asynchronous input through the GPIO peripheral. See Section 15.1.2 to determine how many trip-zone pins are available in the device.
- **Time-base synchronization input (EPWMxSYNCl) and output (EPWMxSYNCO) signals.** The synchronization signals daisy chain the ePWM modules together. Each module can be configured to either use or ignore its synchronization input. The clock synchronization input and output signal are brought out to pins only for ePWM0 (ePWM module #0). The synchronization output for ePWM2 (EPWM2SYNCO) is also connected to the SYNCl of the first enhanced capture module (eCAP0).
- **Peripheral Bus.** The peripheral bus is 32-bits wide and allows both 16-bit and 32-bit writes to the ePWM register file.

Figure 15-9 also shows the key internal submodule interconnect signals. Each submodule is described in Section 15.2.2.

Figure 15-9. ePWM Submodules and Critical Internal Signal Interconnects



15.2.2 Functional Description

Seven submodules are included in every ePWM peripheral. There are some instances that include a high-resolution submodule that allows more precise control of the PWM outputs. Each of these submodules performs specific tasks that can be configured by software.

15.2.2.1 Overview

Table 15-10 lists the eight key submodules together with a list of their main configuration parameters. For example, if you need to adjust or control the duty cycle of a PWM waveform, then you should see the counter-compare submodule in Section 15.2.2.4 for relevant details.

Table 15-10. Submodule Configuration Parameters

Submodule	Configuration Parameter or Option
Time-base (TB)	<ul style="list-style-type: none"> • Scale the time-base clock (TBCLK) relative to the system clock (SYSCLKOUT). • Configure the PWM time-base counter (TBCNT) frequency or period. • Set the mode for the time-base counter: <ul style="list-style-type: none"> – count-up mode: used for asymmetric PWM – count-down mode: used for asymmetric PWM – count-up-and-down mode: used for symmetric PWM • Configure the time-base phase relative to another ePWM module. • Synchronize the time-base counter between modules through hardware or software. • Configure the direction (up or down) of the time-base counter after a synchronization event. • Configure how the time-base counter will behave when the device is halted by an emulator. • Specify the source for the synchronization output of the ePWM module: <ul style="list-style-type: none"> – Synchronization input signal – Time-base counter equal to zero – Time-base counter equal to counter-compare B (CMPB) – No output synchronization signal generated.
Counter-compare (CC)	<ul style="list-style-type: none"> • Specify the PWM duty cycle for output EPWMxA and/or output EPWMxB • Specify the time at which switching events occur on the EPWMxA or EPWMxB output
Action-qualifier (AQ)	<ul style="list-style-type: none"> • Specify the type of action taken when a time-base or counter-compare submodule event occurs: <ul style="list-style-type: none"> – No action taken – Output EPWMxA and/or EPWMxB switched high – Output EPWMxA and/or EPWMxB switched low – Output EPWMxA and/or EPWMxB toggled • Force the PWM output state through software control • Configure and control the PWM dead-band through software
Dead-band (DB)	<ul style="list-style-type: none"> • Control of traditional complementary dead-band relationship between upper and lower switches • Specify the output rising-edge-delay value • Specify the output falling-edge delay value • Bypass the dead-band module entirely. In this case the PWM waveform is passed through without modification.
PWM-chopper (PC)	<ul style="list-style-type: none"> • Create a chopping (carrier) frequency. • Pulse width of the first pulse in the chopped pulse train. • Duty cycle of the second and subsequent pulses. • Bypass the PWM-chopper module entirely. In this case the PWM waveform is passed through without modification.

Table 15-10. Submodule Configuration Parameters (continued)

Submodule	Configuration Parameter or Option
Trip-zone (TZ)	<ul style="list-style-type: none"> • Configure the ePWM module to react to one, all, or none of the trip-zone pins. • Specify the tripping action taken when a fault occurs: <ul style="list-style-type: none"> – Force EPWMxA and/or EPWMxB high – Force EPWMxA and/or EPWMxB low – Force EPWMxA and/or EPWMxB to a high-impedance state – Configure EPWMxA and/or EPWMxB to ignore any trip condition. • Configure how often the ePWM will react to each trip-zone pin: <ul style="list-style-type: none"> – One-shot – Cycle-by-cycle • Enable the trip-zone to initiate an interrupt. • Bypass the trip-zone module entirely.
Event-trigger (ET)	<ul style="list-style-type: none"> • Enable the ePWM events that will trigger an interrupt. • Specify the rate at which events cause triggers (every occurrence or every second or third occurrence) • Poll, set, or clear event flags
High-Resolution PWM (HRPWM)	<ul style="list-style-type: none"> • Enable extended time resolution capabilities • Configure finer time granularity control or edge positioning

Code examples are provided in the remainder of this chapter that show how to implement various ePWM module configurations. These examples use the constant definitions shown in [Example 15-1](#).

Example 15-1. Constant Definitions Used in the Code Examples

```
// TBCTL (Time-Base Control)
// = = = = =
// TBCNT MODE bits
#define TB_COUNT_UP          0x0
#define TB_COUNT_DOWN        0x1
#define TB_COUNT_UPDOWN     0x2
#define TB_FREEZE            0x3
// PHSEN bit
#define TB_DISABLE           0x0
#define TB_ENABLE            0x1
// PRDL bit
#define TB_SHADOW            0x0
#define TB_IMMEDIATE         0x1
// SYNCSEL bits
#define TB_SYNC_IN           0x0
#define TB_CTR_ZERO          0x1
#define TB_CTR_CMPB          0x2
#define TB_SYNC_DISABLE      0x3
// HSPCLKDIV and CLKDIV bits
#define TB_DIV1               0x0
#define TB_DIV2               0x1
#define TB_DIV4               0x2
// PHSDIR bit
#define TB_DOWN               0x0
#define TB_UP                 0x1

// CMPCTL (Compare Control)
// = = = = =
// LOADAMODE and LOADBMODE bits
#define CC_CTR_ZERO          0x0
#define CC_CTR_PRD           0x1
#define CC_CTR_ZERO_PRD     0x2
#define CC_LD_DISABLE        0x3
// SHDWAMODE and SHDWBMODE bits
```


Example 15-1. Constant Definitions Used in the Code Examples (continued)

```

#define          CC_SHADOW          0x0
#define          CC_IMMEDIATE       0x1
// AQCTLA and AQCTLB (Action-qualifier Control)
// = = = = =
// ZRO, PRD, CAU, CAD, CBU, CBD bits
#define          AQ_NO_ACTION       0x0
#define          AQ_CLEAR           0x1
#define          AQ_SET             0x2
#define          AQ_TOGGLE         0x3
// DBCTL (Dead-Band Control)
// = = = = =
// MODE bits
#define          DB_DISABLE         0x0
#define          DBA_ENABLE         0x1
#define          DBB_ENABLE         0x2
#define          DB_FULL_ENABLE     0x3
// POLSEL bits
#define          DB_ACTV_HI         0x0
#define          DB_ACTV_LO        0x1
#define          DB_ACTV_HIC        0x2
#define          DB_ACTV_LO        0x3
// PCCTL (chopper control)
// = = = = =
// CHPEN bit
#define          CHP_ENABLE         0x0
#define          CHP_DISABLE        0x1
// CHPFREQ bits
#define          CHP_DIV1           0x0
#define          CHP_DIV2           0x1
#define          CHP_DIV3           0x2
#define          CHP_DIV4           0x3
#define          CHP_DIV5           0x4
#define          CHP_DIV6           0x5
#define          CHP_DIV7           0x6
#define          CHP_DIV8           0x7
// CHPDUTY bits
#define          CHP1_8TH           0x0
#define          CHP2_8TH           0x1
#define          CHP3_8TH           0x2
#define          CHP4_8TH           0x3
#define          CHP5_8TH           0x4
#define          CHP6_8TH           0x5
#define          CHP7_8TH           0x6
// TZSEL (Trip-zone Select)
// = = = = =
// CBCn and OSHn bits
#define          TZ_ENABLE          0x0
#define          TZ_DISABLE         0x1
// TZCTL (Trip-zone Control)
// = = = = =
// TZA and TZB bits
#define          TZ_HIZ             0x0
#define          TZ_FORCE_HI        0x1
#define          TZ_FORCE_LO        0x2
#define          TZ_DISABLE         0x3
// ETSEL (Event-trigger Select)
// = = = = =
// INTSEL, SOCASEL, SOCBSEL bits
#define          ET_CTR_ZERO        0x1
#define          ET_CTR_PRD         0x2
#define          ET_CTRU_CMPA       0x4
#define          ET_CTRD_CMPA       0x5
#define          ET_CTRU_CMPB       0x6
#define          ET_CTRD_CMPB       0x7
    
```

Example 15-1. Constant Definitions Used in the Code Examples (continued)

```
// ETPS (Event-trigger Prescale)
// =====
// INTPRD, SOCAPRD, SOCBPRD bits
#define      ET_DISABLE      0x0
#define      ET_1ST         0x1
#define      ET_2ND         0x2
#define      ET_3RD         0x3
```

15.2.2.2 Proper Interrupt Initialization Procedure

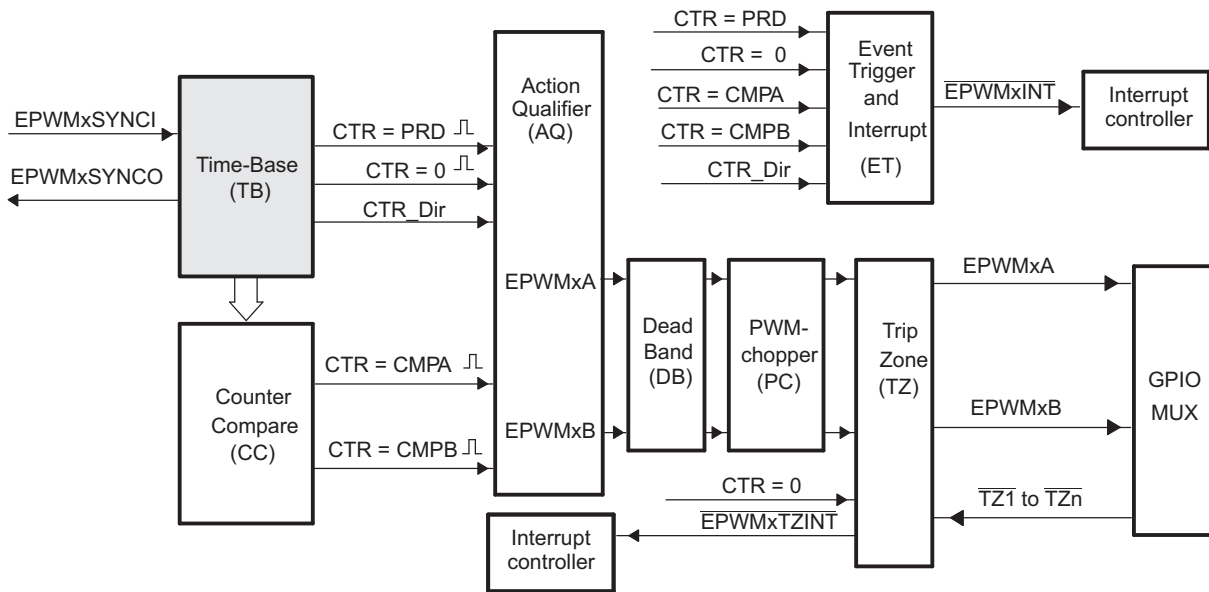
When the ePWM peripheral clock is enabled it may be possible that interrupt flags may be set due to spurious events due to the ePWM registers not being properly initialized. The proper procedure for initializing the ePWM peripheral is:

1. Disable global interrupts (CPU INTM flag)
2. Disable ePWM interrupts
3. Initialize peripheral registers
4. Clear any spurious ePWM flags
5. Enable ePWM interrupts
6. Enable global interrupts

15.2.2.3 Time-Base (TB) Submodule

Each ePWM module has its own time-base submodule that determines all of the event timing for the ePWM module. Built-in synchronization logic allows the time-base of multiple ePWM modules to work together as a single system. Figure 15-10 illustrates the time-base module's place within the ePWM.

Figure 15-10. Time-Base Submodule Block Diagram



15.2.2.3.1 Purpose of the Time-Base Submodule

You can configure the time-base submodule for the following:

- Specify the ePWM time-base counter (TBCNT) frequency or period to control how often events occur.
- Manage time-base synchronization with other ePWM modules.
- Maintain a phase relationship with other ePWM modules.
- Set the time-base counter to count-up, count-down, or count-up-and-down mode.
- Generate the following events:
 - CTR = PRD: Time-base counter equal to the specified period (TBCNT = TBPRD) .
 - CTR = 0: Time-base counter equal to zero (TBCNT = 0000h).
- Configure the rate of the time-base clock; a prescaled version of the CPU system clock (SYSCLKOUT). This allows the time-base counter to increment/decrement at a slower rate.

15.2.2.3.2 Controlling and Monitoring the Time-Base Submodule

Table 15-11 lists the registers used to control and monitor the time-base submodule.

Table 15-11. Time-Base Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
TBCTL	Time-Base Control Register	0h	No
TBSTS	Time-Base Status Register	2h	No
TBPHSHR	HRPWM extension Phase Register ⁽¹⁾	4h	No
TBPHS	Time-Base Phase Register	6h	No
TBCNT	Time-Base Counter Register	8h	No
TBPRD	Time-Base Period Register	Ah	Yes

⁽¹⁾ This register is available only on ePWM instances that include the high-resolution extension (HRPWM). On ePWM modules that do not include the HRPWM, this location is reserved. See Section 15.1.2 to determine which ePWM instances include this feature.

Figure 15-11 shows the critical signals and registers of the time-base submodule. Table 15-12 provides descriptions of the key signals associated with the time-base submodule.

Figure 15-11. Time-Base Submodule Signals and Registers

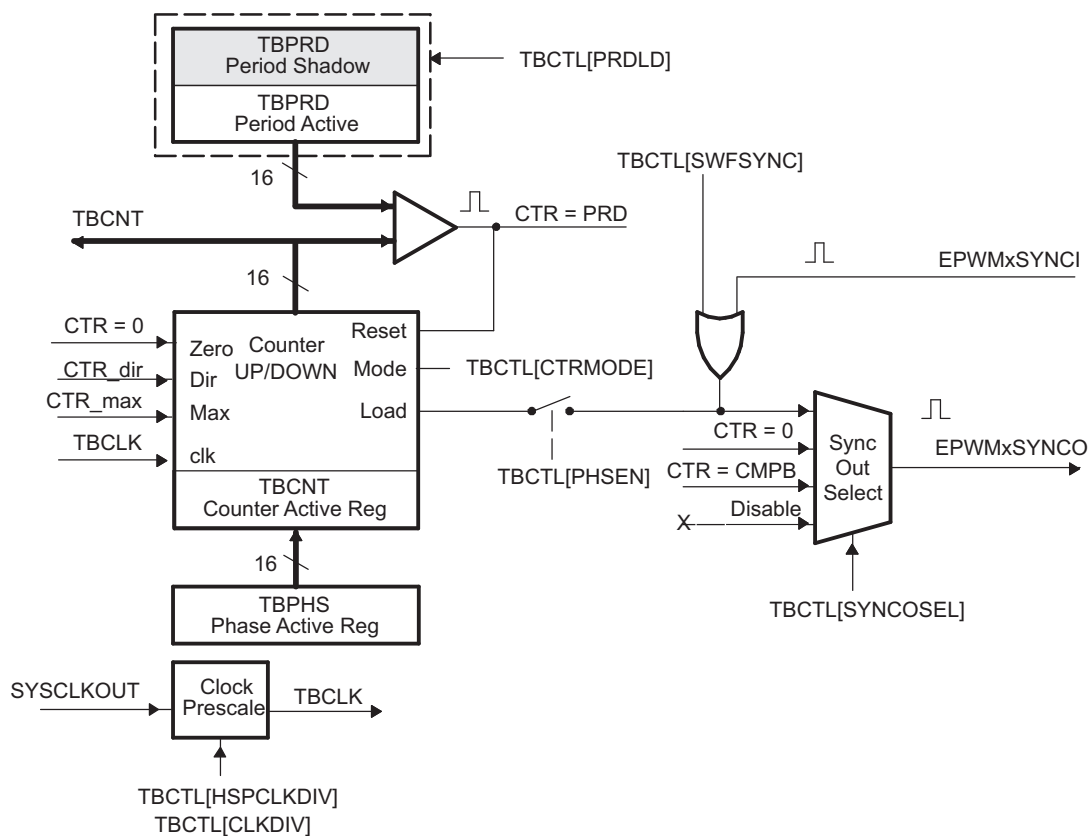


Table 15-12. Key Time-Base Signals

Signal	Description
EPWMxSYNCl	Time-base synchronization input. Input pulse used to synchronize the time-base counter with the counter of ePWM module earlier in the synchronization chain. An ePWM peripheral can be configured to use or ignore this signal. For example, this signal could come from a device pin for the first ePWM module (ePWM0). For subsequent ePWM modules this signal could be passed from another ePWM peripheral, such that EPWM1SYNCl is generated by the ePWM0 peripheral, EPWM2SYNCl is generated by ePWM1, and so forth. See Section 15.1.2 for information on the synchronization order of a particular device.
EPWMxSYNCO	Time-base synchronization output. This output pulse is used to synchronize the counter of an ePWM module later in the synchronization chain. The ePWM module generates this signal from one of three event sources: <ol style="list-style-type: none"> 1. EPWMxSYNCl (Synchronization input pulse) 2. CTR = 0: The time-base counter equal to zero (TBCNT = 0000h). 3. CTR = CMPB: The time-base counter equal to the counter-compare B (TBCNT = CMPB) register.
CTR = PRD	Time-base counter equal to the specified period. This signal is generated whenever the counter value is equal to the active period register value. That is when TBCNT = TBPRD.
CTR = 0	Time-base counter equal to zero. This signal is generated whenever the counter value is zero. That is when TBCNT equals 0000h.
CTR = CMPB	Time-base counter equal to active counter-compare B register (TBCNT = CMPB). This event is generated by the counter-compare submodule and used by the synchronization out logic.
CTR_dir	Time-base counter direction. Indicates the current direction of the ePWM's time-base counter. This signal is high when the counter is increasing and low when it is decreasing.
CTR_max	Time-base counter equal max value. (TBCNT = FFFFh) Generated event when the TBCNT value reaches its maximum value. This signal is only used only as a status bit.
TBCLK	Time-base clock. This is a prescaled version of the system clock (SYSCLKOUT) and is used by all submodules within the ePWM. This clock determines the rate at which time-base counter increments or decrements.

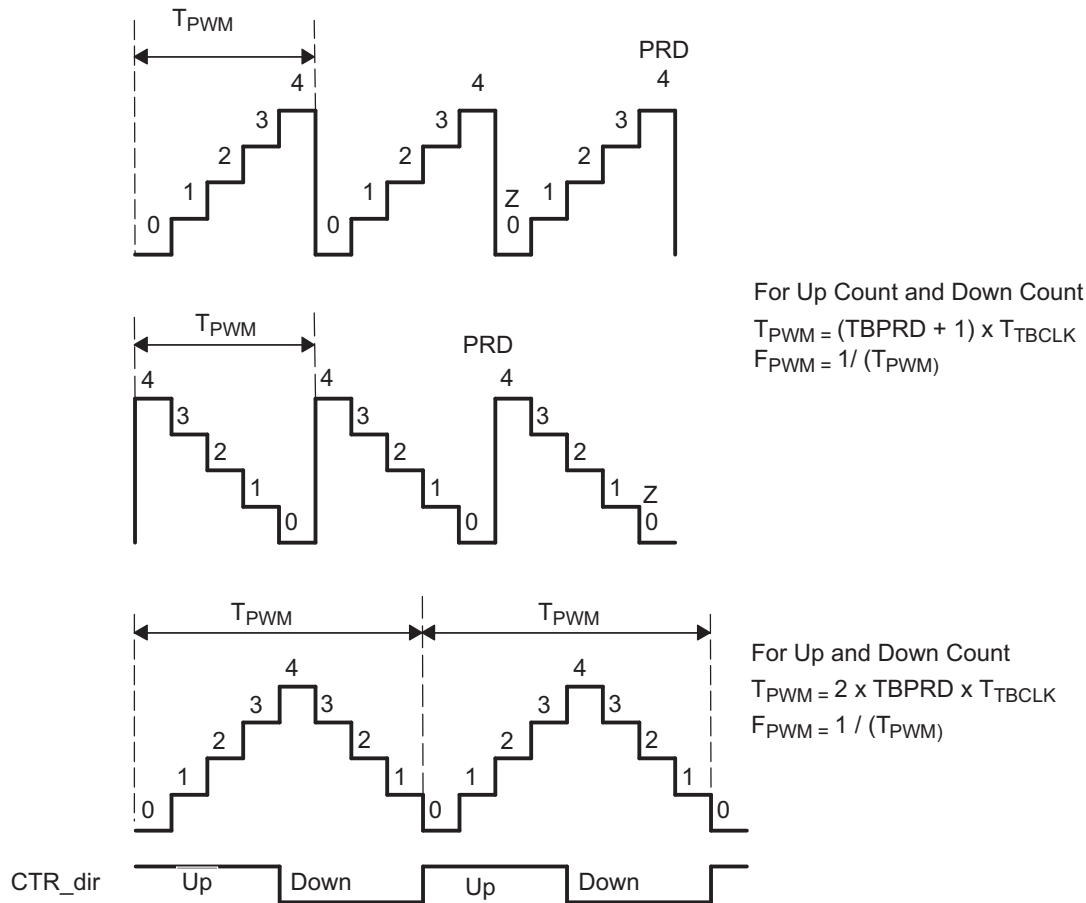
15.2.2.3.3 Calculating PWM Period and Frequency

The frequency of PWM events is controlled by the time-base period (TBPRD) register and the mode of the time-base counter. [Figure 15-12](#) shows the period (T_{pwm}) and frequency (F_{pwm}) relationships for the up-count, down-count, and up-down-count time-base counter modes when the period is set to 4 (TBPRD = 4). The time increment for each step is defined by the time-base clock (TBCLK) which is a prescaled version of the system clock (SYSCLKOUT).

The time-base counter has three modes of operation selected by the time-base control register (TBCTL):

- **Up-Down-Count Mode:** In up-down-count mode, the time-base counter starts from zero and increments until the period (TBPRD) value is reached. When the period value is reached, the time-base counter then decrements until it reaches zero. At this point the counter repeats the pattern and begins to increment.
- **Up-Count Mode:** In this mode, the time-base counter starts from zero and increments until it reaches the value in the period register (TBPRD). When the period value is reached, the time-base counter resets to zero and begins to increment once again.
- **Down-Count Mode:** In down-count mode, the time-base counter starts from the period (TBPRD) value and decrements until it reaches zero. When it reaches zero, the time-base counter is reset to the period value and it begins to decrement once again.

Figure 15-12. Time-Base Frequency and Period



15.2.2.3.3.1 Time-Base Period Shadow Register

The time-base period register (TBPRD) has a shadow register. Shadowing allows the register update to be synchronized with the hardware. The following definitions are used to describe all shadow registers in the ePWM module:

- **Active Register:** The active register controls the hardware and is responsible for actions that the hardware causes or invokes.
- **Shadow Register:** The shadow register buffers or provides a temporary holding location for the active register. It has no direct effect on any control hardware. At a strategic point in time the shadow register's content is transferred to the active register. This prevents corruption or spurious operation due to the register being asynchronously modified by software.

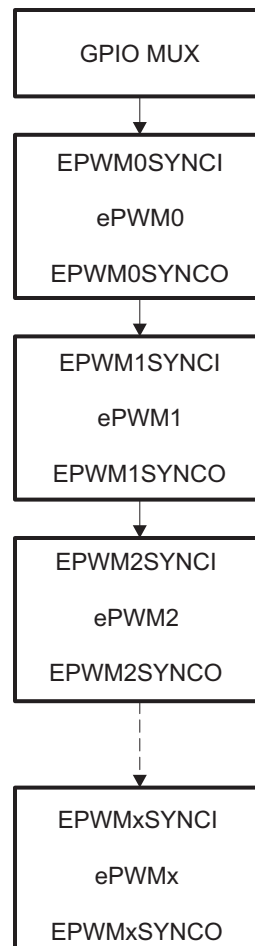
The memory address of the shadow period register is the same as the active register. Which register is written to or read from is determined by the TBCTL[PRDL] bit. This bit enables and disables the TBPRD shadow register as follows:

- **Time-Base Period Shadow Mode:** The TBPRD shadow register is enabled when TBCTL[PRDL] = 0. Reads from and writes to the TBPRD memory address go to the shadow register. The shadow register contents are transferred to the active register (TBPRD (Active) ← TBPRD (shadow)) when the time-base counter equals zero (TBCNT = 0000h). By default the TBPRD shadow register is enabled.
- **Time-Base Period Immediate Load Mode:** If immediate load mode is selected (TBCTL[PRDL] = 1), then a read from or a write to the TBPRD memory address goes directly to the active register.

15.2.2.3.3.2 Time-Base Counter Synchronization

A time-base synchronization scheme connects all of the ePWM modules on a device. Each ePWM module has a synchronization input (EPWMxSYNCl) and a synchronization output (EPWMxSYNCO). The synchronization input can come from an external pin or another ePWM module. An example of synchronization connections for the remaining ePWM modules is shown in [Section 15.1.2](#).

Figure 15-13. Time-Base Counter Synchronization Scheme 1



Each ePWM module can be configured to use or ignore the synchronization input. If the TBCTL[PHSEN] bit is set, then the time-base counter (TBCNT) of the ePWM module will be automatically loaded with the phase register (TBPHS) contents when one of the following conditions occur:

- **EPWMxSYNCl: Synchronization Input Pulse:** The value of the phase register is loaded into the counter register when an input synchronization pulse is detected (TBPHS → TBCNT). This operation occurs on the next valid time-base clock (TBCLK) edge.
- **Software Forced Synchronization Pulse:** Writing a 1 to the TBCTL[SWFSYNC] control bit invokes a software forced synchronization. This pulse is ORed with the synchronization input signal, and therefore has the same effect as a pulse on EPWMxSYNCl.

This feature enables the ePWM module to be automatically synchronized to the time base of another ePWM module. Lead or lag phase control can be added to the waveforms generated by different ePWM modules to synchronize them. In up-down-count mode, the TBCTL[PSHDIR] bit configures the direction of the time-base counter immediately after a synchronization event. The new direction is independent of the direction prior to the synchronization event. The TBPHS bit is ignored in count-up or count-down modes. See [Figure 15-14](#) through [Figure 15-17](#) for examples.

Clearing the TBCTL[PHSEN] bit configures the ePWM to ignore the synchronization input pulse. The synchronization pulse can still be allowed to flow-through to the EPWMxSYNCO and be used to synchronize other ePWM modules. In this way, you can set up a master time-base (for example, ePWM0) and downstream modules (ePWM1 – ePWMx) may elect to run in synchronization with the master.

15.2.2.3.4 Phase Locking the Time-Base Clocks of Multiple ePWM Modules

The TBCLKEN bit in the PWMSS_CTRL register in the Control Module can be used to globally synchronize the time-base clocks of all enabled ePWM modules on a device. The TBCLKEN bit is part of the chip configuration registers and is described in [Chapter 9](#). When TBCLKEN = 0, the time-base clock of all ePWM modules is stopped (default). When TBCLKEN = 1, all ePWM time-base clocks are started with the rising edge of TBCLK aligned. For perfectly synchronized TBCLKs, the prescaler bits in the TBCTL register of each ePWM module must be set identically. The proper procedure for enabling the ePWM clocks is as follows:

1. Enable the ePWM module clocks.
2. Set TBCLKEN = 0. This will stop the time-base clock within any enabled ePWM module.
3. Configure the prescaler values and desired ePWM modes.
4. Set TBCLKEN = 1.

15.2.2.3.5 Time-Base Counter Modes and Timing Waveforms

The time-base counter operates in one of four modes:

- Up-count mode which is asymmetrical.
- Down-count mode which is asymmetrical.
- Up-down-count which is symmetrical.
- Frozen where the time-base counter is held constant at the current value.

To illustrate the operation of the first three modes, [Figure 15-14](#) to [Figure 15-17](#) show when events are generated and how the time-base responds to an EPWMxSYNCl signal.

Figure 15-14. Time-Base Up-Count Mode Waveforms

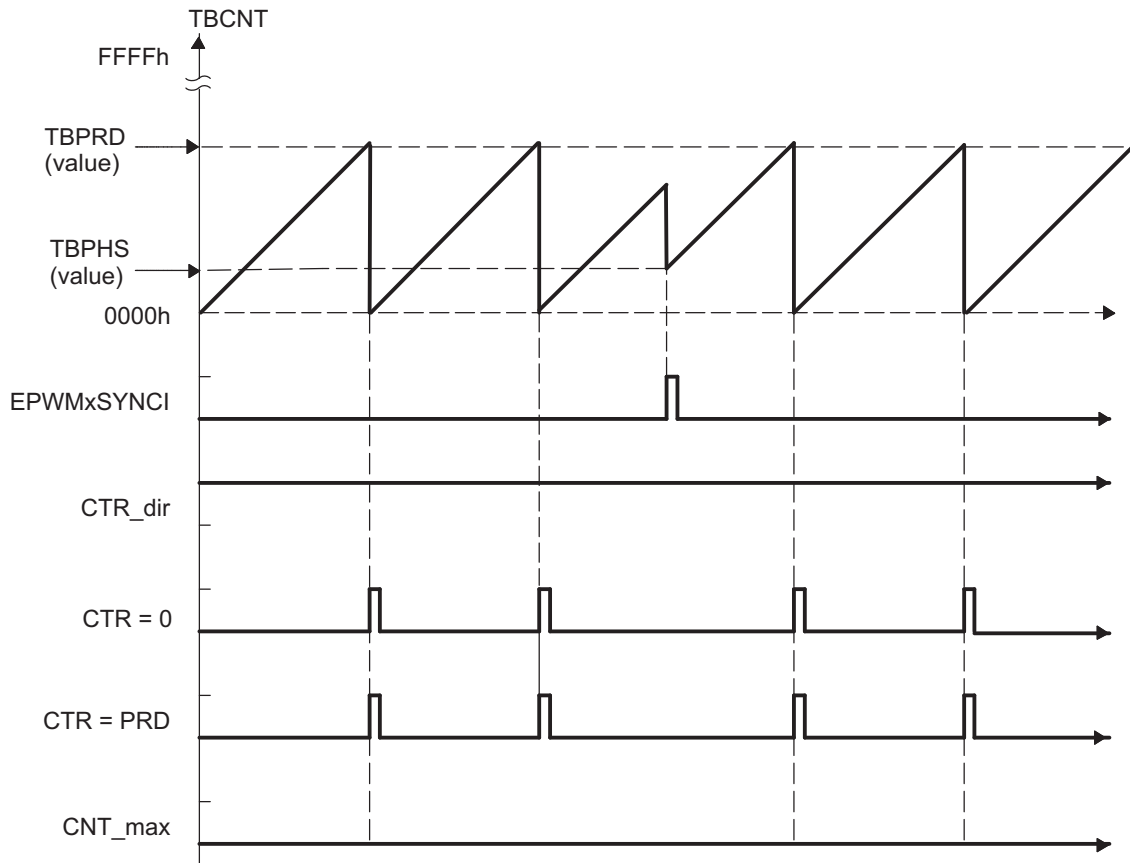


Figure 15-15. Time-Base Down-Count Mode Waveforms

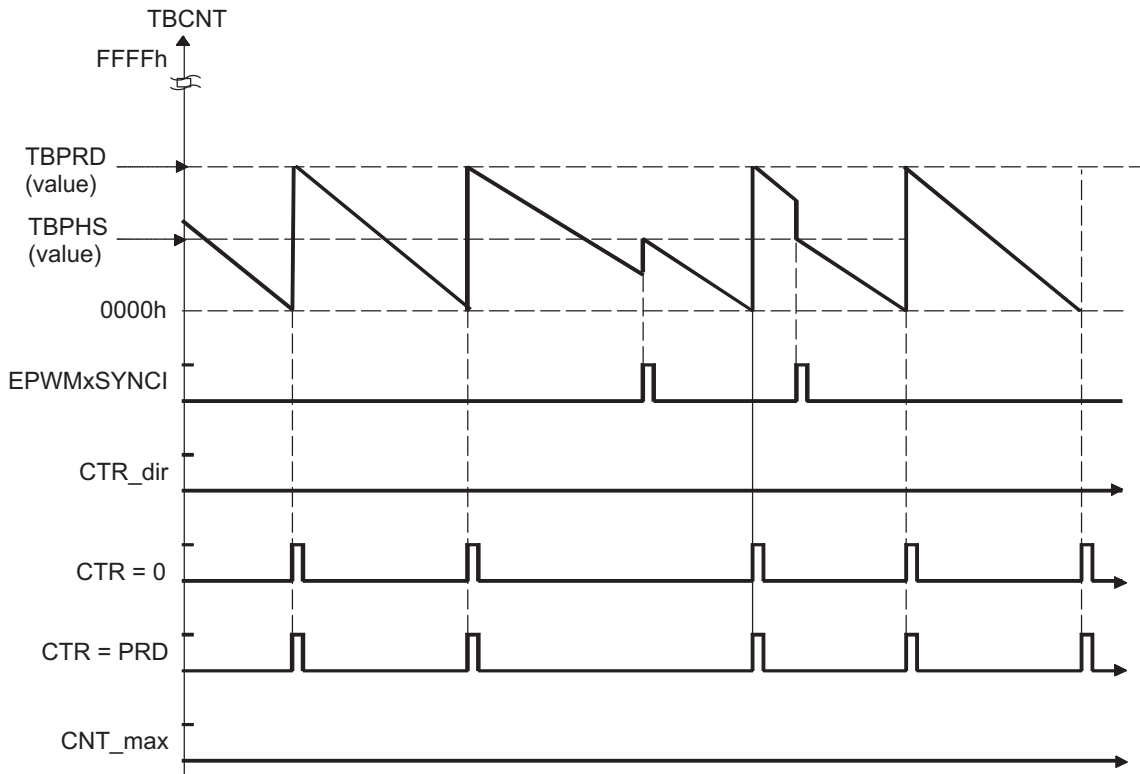


Figure 15-16. Time-Base Up-Down-Count Waveforms, TBCTL[PHSDIR = 0] Count Down on Synchronization Event

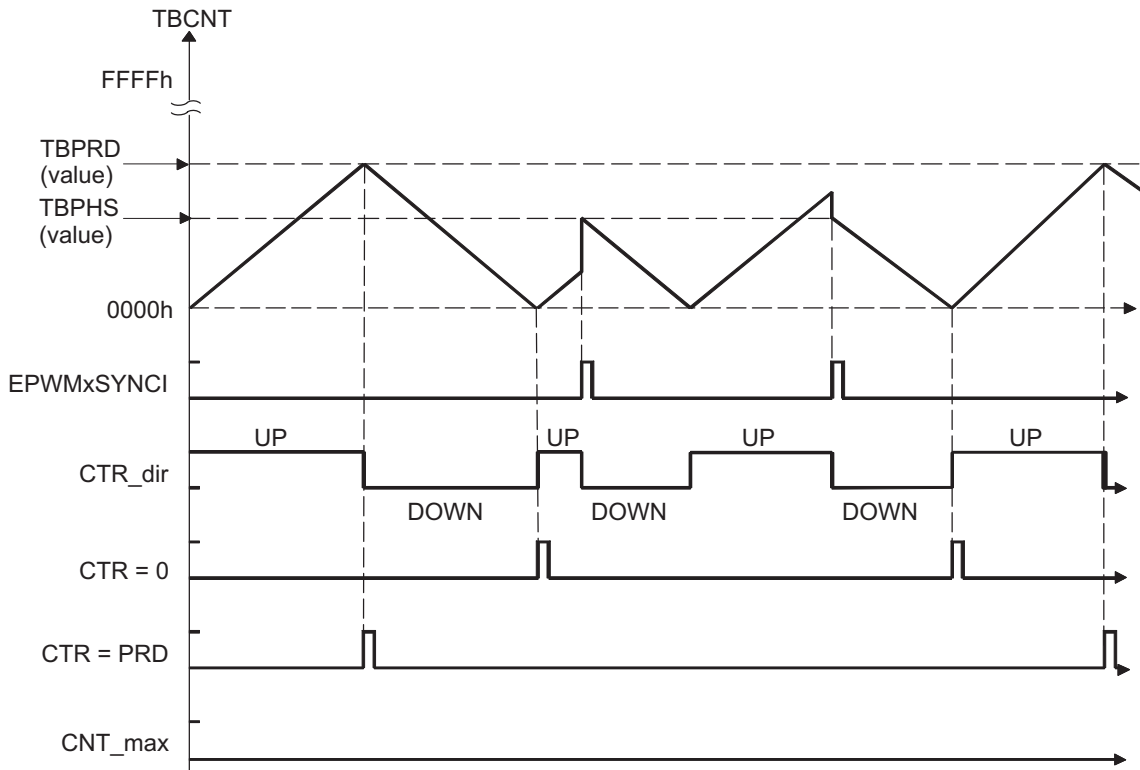
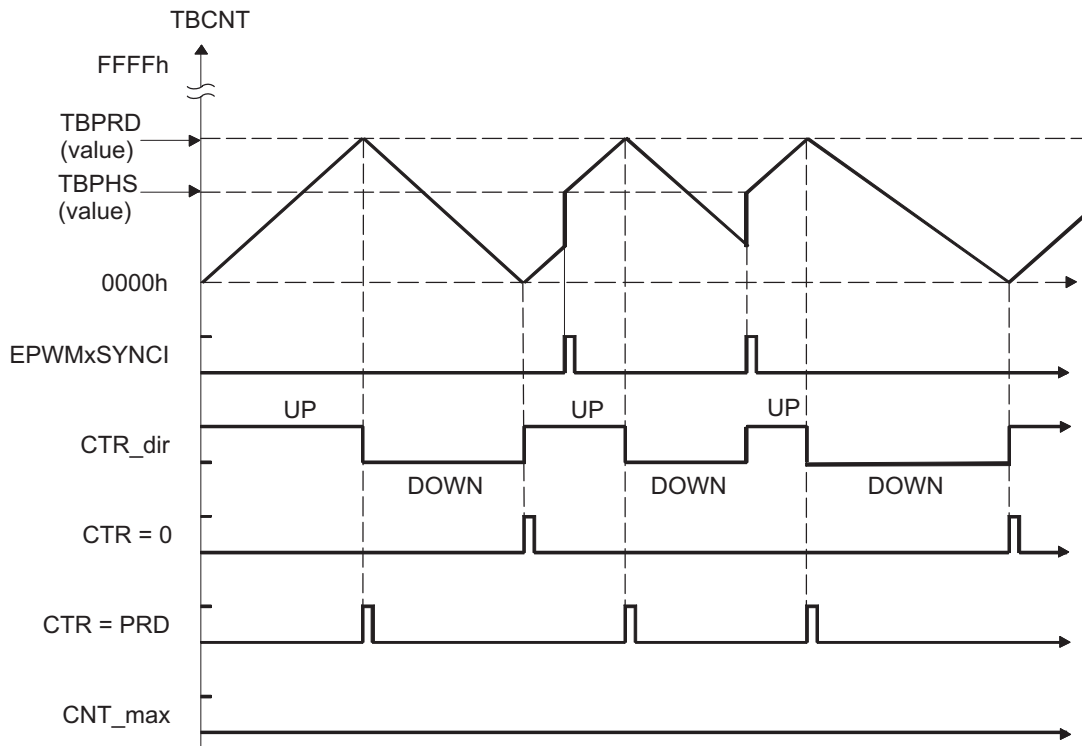


Figure 15-17. Time-Base Up-Down Count Waveforms, TBCTL[PHSDIR = 1] Count Up on Synchronization Event



15.2.2.4 Counter-Compare (CC) Submodule

Figure 15-18 illustrates the counter-compare submodule within the ePWM. Figure 15-19 shows the basic structure of the counter-compare submodule.

Figure 15-18. Counter-Compare Submodule

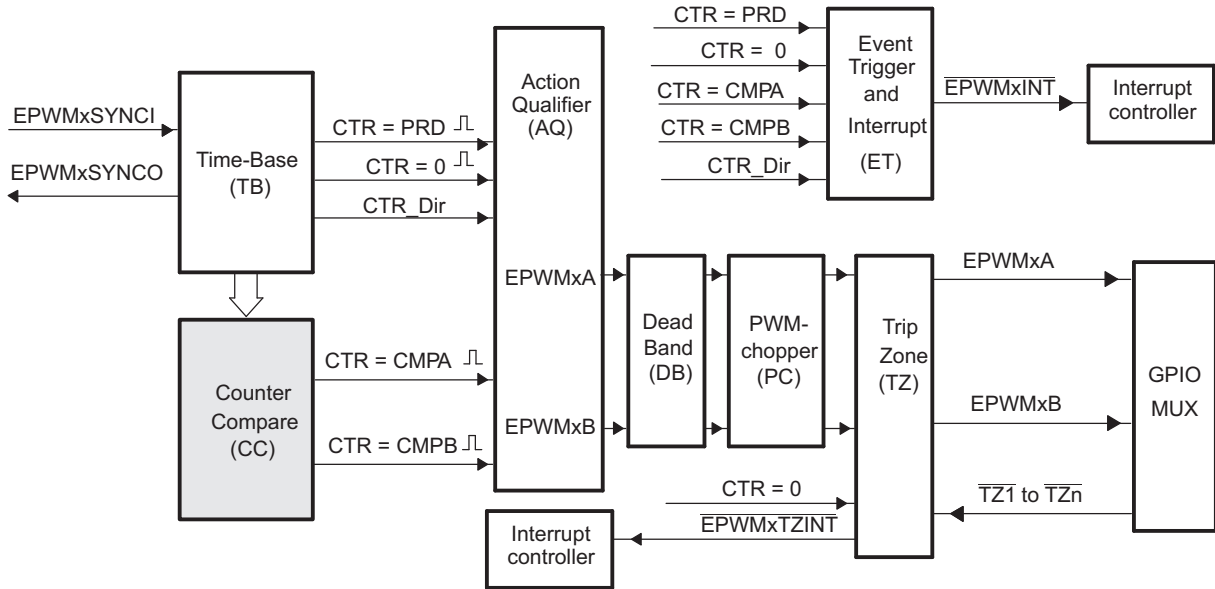
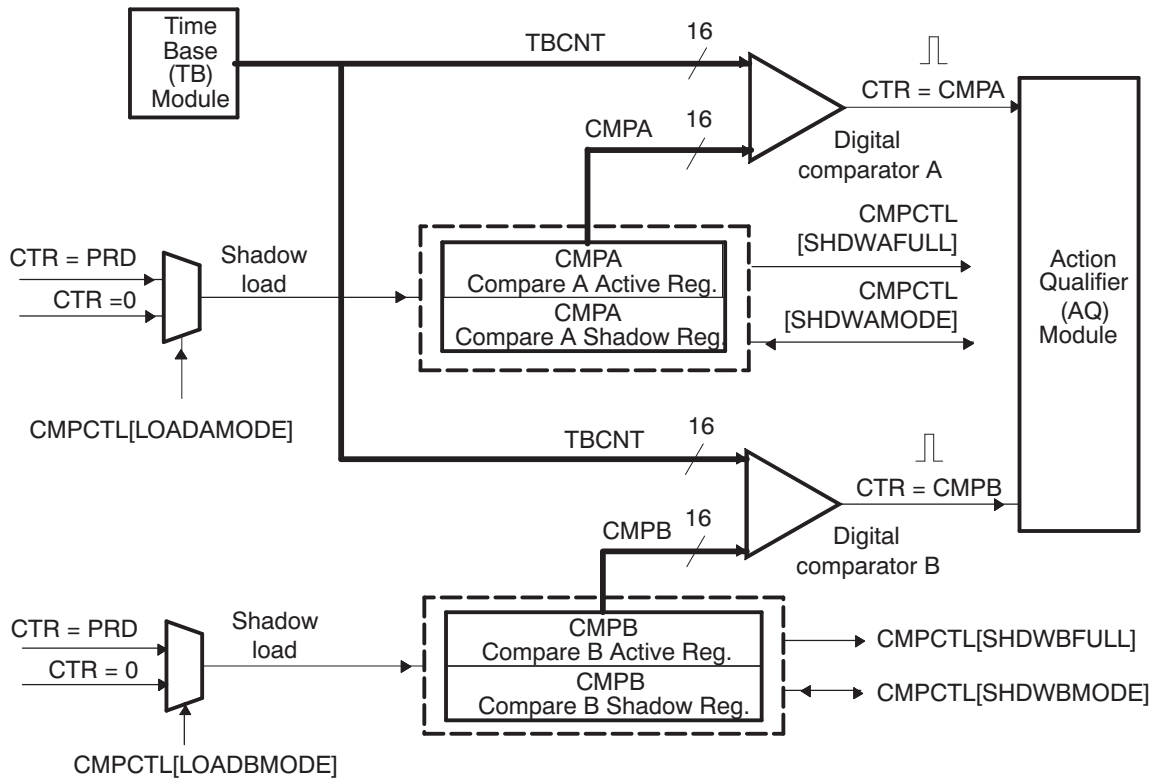


Figure 15-19. Counter-Compare Submodule Signals and Registers



15.2.2.4.1 Purpose of the Counter-Compare Submodule

The counter-compare submodule takes as input the time-base counter value. This value is continuously compared to the counter-compare A (CMPA) and counter-compare B (CMPB) registers. When the time-base counter is equal to one of the compare registers, the counter-compare unit generates an appropriate event.

The counter-compare submodule:

- Generates events based on programmable time stamps using the CMPA and CMPB registers
 - CTR = CMPA: Time-base counter equals counter-compare A register (TBCNT = CMPA).
 - CTR = CMPB: Time-base counter equals counter-compare B register (TBCNT = CMPB)
- Controls the PWM duty cycle if the action-qualifier submodule is configured appropriately
- Shadows new compare values to prevent corruption or glitches during the active PWM cycle

15.2.2.4.2 Controlling and Monitoring the Counter-Compare Submodule

Table 15-13 lists the registers used to control and monitor the counter-compare submodule. Table 15-14 lists the key signals associated with the counter-compare submodule.

Table 15-13. Counter-Compare Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
CMPCTL	Counter-Compare Control Register.	Eh	No
CMPAHR	HRPWM Counter-Compare A Extension Register ⁽¹⁾	10h	Yes
CMPA	Counter-Compare A Register	12h	Yes
CMPB	Counter-Compare B Register	14h	Yes

⁽¹⁾ This register is available only on ePWM modules with the high-resolution extension (HRPWM). On ePWM modules that do not include the HRPWM, this location is reserved. See Section 15.1.2 to determine which ePWM instances include this feature.

Table 15-14. Counter-Compare Submodule Key Signals

Signal	Description of Event	Registers Compared
CTR = CMPA	Time-base counter equal to the active counter-compare A value	TBCNT = CMPA
CTR = CMPB	Time-base counter equal to the active counter-compare B value	TBCNT = CMPB
CTR = PRD	Time-base counter equal to the active period. Used to load active counter-compare A and B registers from the shadow register	TBCNT = TBPRD
CTR = 0	Time-base counter equal to zero. Used to load active counter-compare A and B registers from the shadow register	TBCNT = 0000h

15.2.2.4.3 Operational Highlights for the Counter-Compare Submodule

The counter-compare submodule is responsible for generating two independent compare events based on two compare registers:

1. CTR = CMPA: Time-base counter equal to counter-compare A register (TBCNT = CMPA).
2. CTR = CMPB: Time-base counter equal to counter-compare B register (TBCNT = CMPB).

For up-count or down-count mode, each event occurs only once per cycle. For up-down-count mode each event occurs twice per cycle, if the compare value is between 0000h and TBPRD; and occurs once per cycle, if the compare value is equal to 0000h or equal to TBPRD. These events are fed into the action-qualifier submodule where they are qualified by the counter direction and converted into actions if enabled. Refer to [Section 15.2.2.5.1](#) for more details.

The counter-compare registers CMPA and CMPB each have an associated shadow register. Shadowing provides a way to keep updates to the registers synchronized with the hardware. When shadowing is used, updates to the active registers only occurs at strategic points. This prevents corruption or spurious operation due to the register being asynchronously modified by software. The memory address of the active register and the shadow register is identical. Which register is written to or read from is determined by the CMPCTL[SHDWAMODE] and CMPCTL[SHDWBMODE] bits. These bits enable and disable the CMPA shadow register and CMPB shadow register respectively. The behavior of the two load modes is described below:

- **Shadow Mode:** The shadow mode for the CMPA is enabled by clearing the CMPCTL[SHDWAMODE] bit and the shadow register for CMPB is enabled by clearing the CMPCTL[SHDWBMODE] bit. Shadow mode is enabled by default for both CMPA and CMPB.

If the shadow register is enabled then the content of the shadow register is transferred to the active register on one of the following events:

- CTR = PRD: Time-base counter equal to the period (TBCNT = TBPRD).
- CTR = 0: Time-base counter equal to zero (TBCNT = 0000h)
- Both CTR = PRD and CTR = 0

Which of these three events is specified by the CMPCTL[LOADAMODE] and CMPCTL[LOADBMODE] register bits. Only the active register contents are used by the counter-compare submodule to generate events to be sent to the action-qualifier.

- **Immediate Load Mode:** If immediate load mode is selected (TBCTL[SHADWAMODE] = 1 or TBCTL[SHADWBMODE] = 1), then a read from or a write to the register will go directly to the active register.

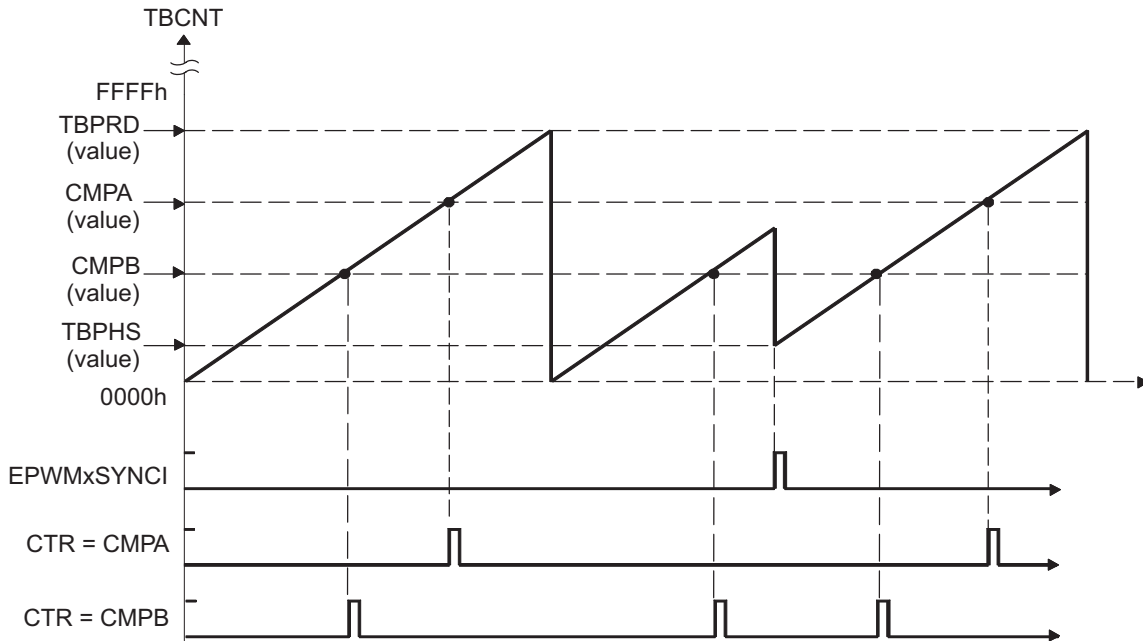
15.2.2.4.4 Count Mode Timing Waveforms

The counter-compare module can generate compare events in all three count modes:

- Up-count mode: used to generate an asymmetrical PWM waveform.
- Down-count mode: used to generate an asymmetrical PWM waveform.
- Up-down-count mode: used to generate a symmetrical PWM waveform.

To best illustrate the operation of the first three modes, the timing diagrams in [Figure 15-20](#) to [Figure 15-23](#) show when events are generated and how the EPWMxSYNCl signal interacts.

Figure 15-20. Counter-Compare Event Waveforms in Up-Count Mode



NOTE: An EPWMxSYNCl external synchronization event can cause a discontinuity in the TBCNT count sequence. This can lead to a compare event being skipped. This skipping is considered normal operation and must be taken into account.

Figure 15-21. Counter-Compare Events in Down-Count Mode

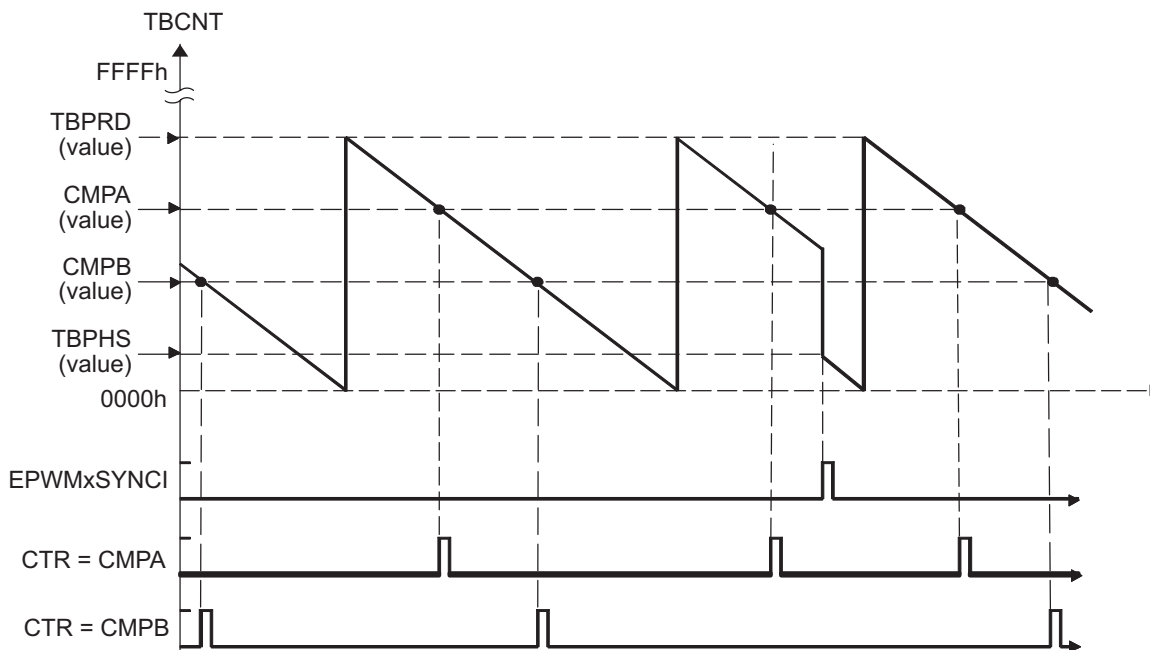


Figure 15-22. Counter-Compare Events in Up-Down-Count Mode, TBCTL[PHSDIR = 0] Count Down on Synchronization Event

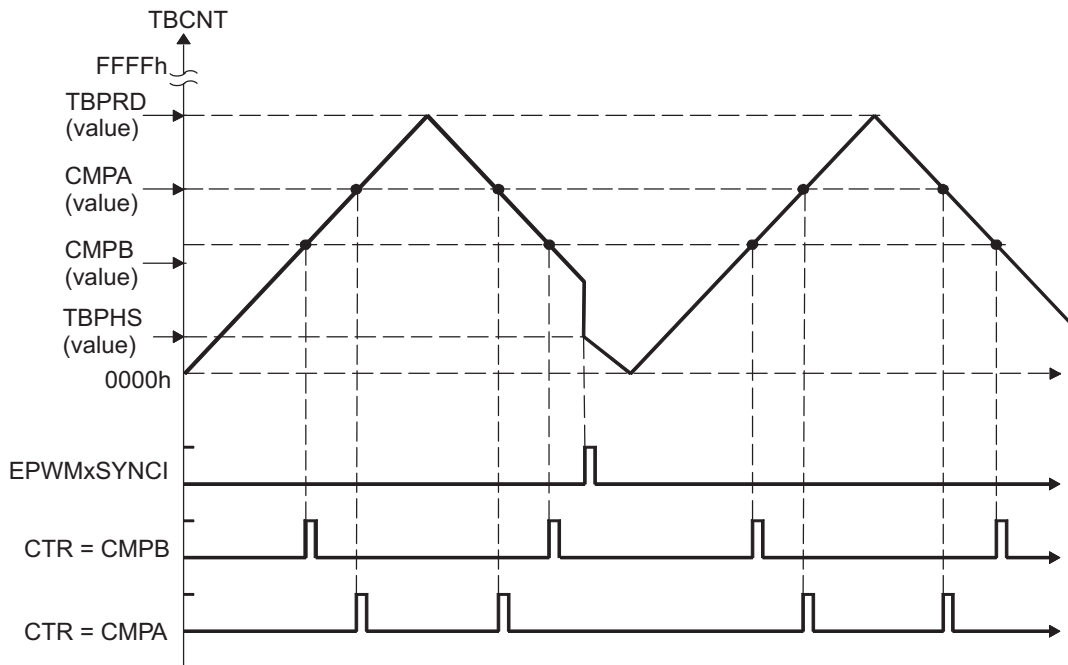
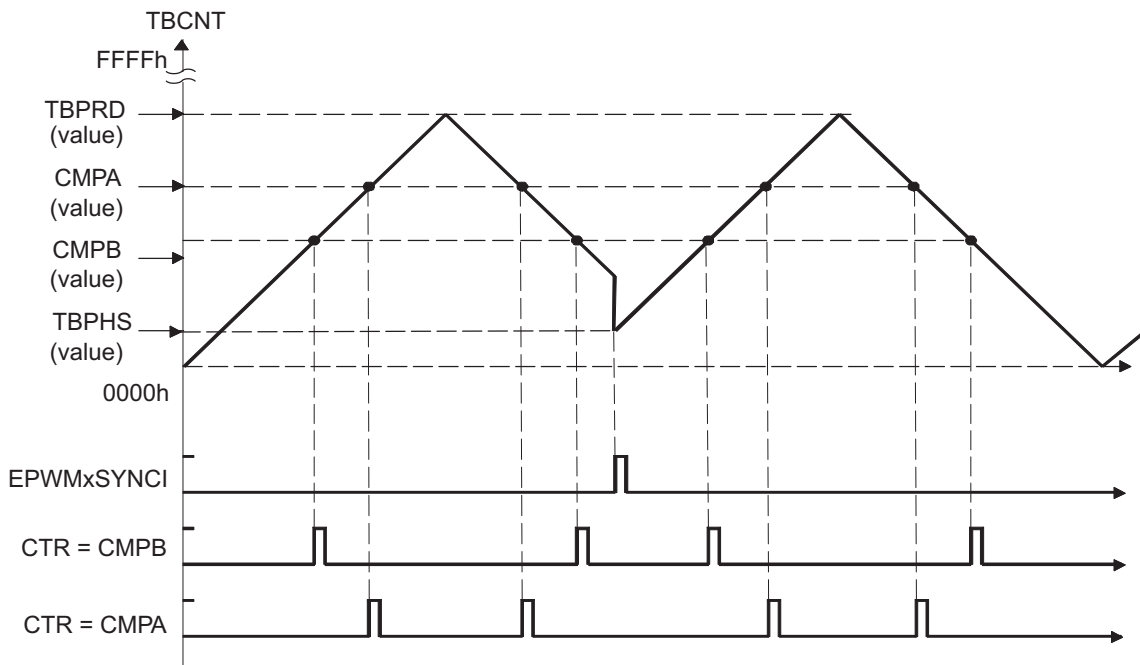


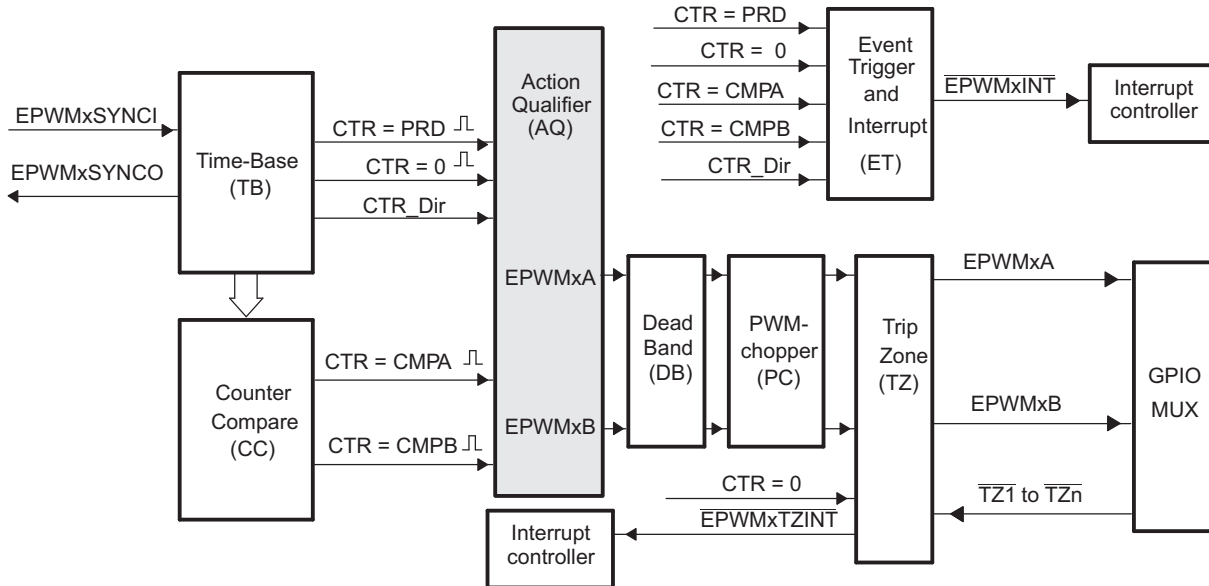
Figure 15-23. Counter-Compare Events in Up-Down-Count Mode, TBCTL[PHSDIR = 1] Count Up on Synchronization Event



15.2.2.5 Action-Qualifier (AQ) Submodule

Figure 15-24 shows the action-qualifier (AQ) submodule (see shaded block) in the ePWM system. The action-qualifier submodule has the most important role in waveform construction and PWM generation. It decides which events are converted into various action types, thereby producing the required switched waveforms at the EPWMxA and EPWMxB outputs.

Figure 15-24. Action-Qualifier Submodule



15.2.2.5.1 Purpose of the Action-Qualifier Submodule

The action-qualifier submodule is responsible for the following:

- Qualifying and generating actions (set, clear, toggle) based on the following events:
 - CTR = PRD: Time-base counter equal to the period (TBCNT = TBPRD)
 - CTR = 0: Time-base counter equal to zero (TBCNT = 0000h)
 - CTR = CMPA: Time-base counter equal to the counter-compare A register (TBCNT = CMPA)
 - CTR = CMPB: Time-base counter equal to the counter-compare B register (TBCNT = CMPB)
- Managing priority when these events occur concurrently
- Providing independent control of events when the time-base counter is increasing and when it is decreasing.

15.2.2.5.2 Controlling and Monitoring the Action-Qualifier Submodule

Table 15-15 lists the registers used to control and monitor the action-qualifier submodule.

Table 15-15. Action-Qualifier Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
AQCTLA	Action-Qualifier Control Register For Output A (EPWMxA)	16h	No
AQCTLB	Action-Qualifier Control Register For Output B (EPWMxB)	18h	No
AQSFRC	Action-Qualifier Software Force Register	1Ah	No
AQCSFRC	Action-Qualifier Continuous Software Force	1Ch	Yes

The action-qualifier submodule is based on event-driven logic. It can be thought of as a programmable cross switch with events at the input and actions at the output, all of which are software controlled via the set of registers shown in Figure 15-25. The possible input events are summarized again in Table 15-16.

Figure 15-25. Action-Qualifier Submodule Inputs and Outputs

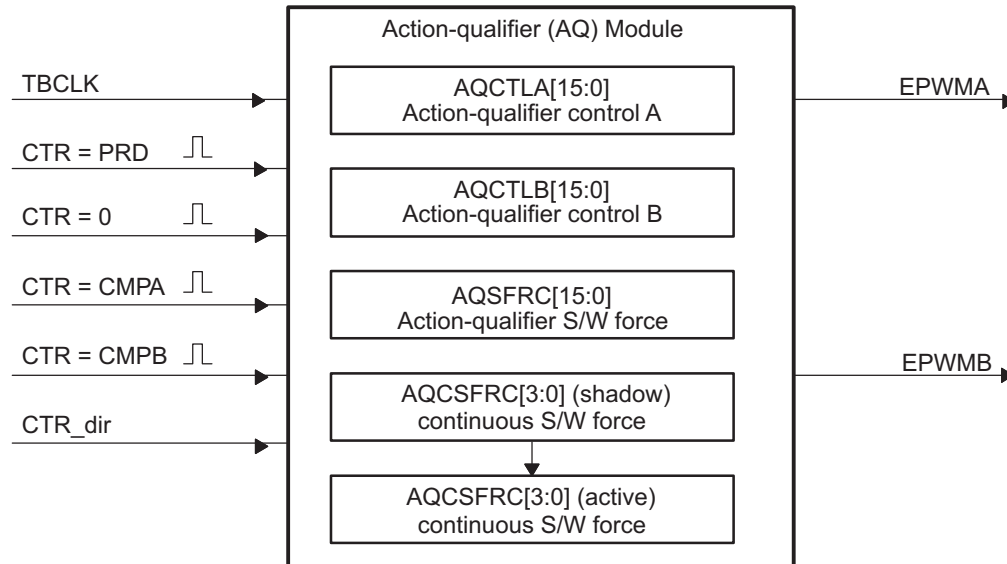


Table 15-16. Action-Qualifier Submodule Possible Input Events

Signal	Description	Registers Compared
CTR = PRD	Time-base counter equal to the period value	TBCNT = TBPRD
CTR = 0	Time-base counter equal to zero	TBCNT = 0000h
CTR = CMPA	Time-base counter equal to the counter-compare A	TBCNT = CMPA
CTR = CMPB	Time-base counter equal to the counter-compare B	TBCNT = CMPB
Software forced event	Asynchronous event initiated by software	

The software forced action is a useful asynchronous event. This control is handled by registers AQSFR and AQCSFR.

The action-qualifier submodule controls how the two outputs EPWMxA and EPWMxB behave when a particular event occurs. The event inputs to the action-qualifier submodule are further qualified by the counter direction (up or down). This allows for independent action on outputs on both the count-up and count-down phases.





















The possible actions imposed on outputs EPWMxA and EPWMxB are:

- **Set High:** Set output EPWMxA or EPWMxB to a high level.
- **Clear Low:** Set output EPWMxA or EPWMxB to a low level.
- **Toggle:** If EPWMxA or EPWMxB is currently pulled high, then pull the output low. If EPWMxA or EPWMxB is currently pulled low, then pull the output high.
- **Do Nothing:** Keep outputs EPWMxA and EPWMxB at same level as currently set. Although the "Do Nothing" option prevents an event from causing an action on the EPWMxA and EPWMxB outputs, this event can still trigger interrupts. See the event-trigger submodule description in Section 15.2.2.9 for details.

Actions are specified independently for either output (EPWMxA or EPWMxB). Any or all events can be configured to generate actions on a given output. For example, both CTR = CMPA and CTR = CMPB can operate on output EPWMxA. All qualifier actions are configured via the control registers found at the end of this section.

For clarity, the drawings in this chapter use a set of symbolic actions. These symbols are summarized in Figure 15-26. Each symbol represents an action as a marker in time. Some actions are fixed in time (zero and period) while the CMPA and CMPB actions are moveable and their time positions are programmed via the counter-compare A and B registers, respectively. To turn off or disable an action, use the "Do Nothing option"; it is the default at reset.

Figure 15-26. Possible Action-Qualifier Actions for EPWMxA and EPWMxB Outputs

S/W force	TB Counter equals:				Actions
	Zero	Comp A	Comp B	Period	
					Do Nothing
					Clear Low
					Set High
					Toggle

15.2.2.5.3 Action-Qualifier Event Priority

It is possible for the ePWM action qualifier to receive more than one event at the same time. In this case events are assigned a priority by the hardware. The general rule is events occurring later in time have a higher priority and software forced events always have the highest priority. The event priority levels for up-down-count mode are shown in [Table 15-17](#). A priority level of 1 is the highest priority and level 7 is the lowest. The priority changes slightly depending on the direction of TBCNT.

Table 15-17. Action-Qualifier Event Priority for Up-Down-Count Mode

Priority Level	Event if TBCNT is Incrementing TBCNT = 0 up to TBCNT = TBPRD	Event if TBCNT is Decrementing TBCNT = TBPRD down to TBCNT = 1
1 (Highest)	Software forced event	Software forced event
2	Counter equals CMPB on up-count (CBU)	Counter equals CMPB on down-count (CBD)
3	Counter equals CMPA on up-count (CAU)	Counter equals CMPA on down-count (CAD)
4	Counter equals zero	Counter equals period (TBPRD)
5	Counter equals CMPB on down-count (CBD) ⁽¹⁾	Counter equals CMPB on up-count (CBU) ⁽¹⁾
6 (Lowest)	Counter equals CMPA on down-count (CAD) ⁽¹⁾	Counter equals CMPA on up-count (CBU) ⁽¹⁾

⁽¹⁾ To maintain symmetry for up-down-count mode, both up-events (CAU/CBU) and down-events (CAD/CBD) can be generated for TBPRD. Otherwise, up-events can occur only when the counter is incrementing and down-events can occur only when the counter is decrementing.

[Table 15-18](#) shows the action-qualifier priority for up-count mode. In this case, the counter direction is always defined as up and thus down-count events will never be taken.

Table 15-18. Action-Qualifier Event Priority for Up-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to period (TBPRD)
3	Counter equal to CMPB on up-count (CBU)
4	Counter equal to CMPA on up-count (CAU)
5 (Lowest)	Counter equal to Zero

[Table 15-19](#) shows the action-qualifier priority for down-count mode. In this case, the counter direction is always defined as down and thus up-count events will never be taken.

Table 15-19. Action-Qualifier Event Priority for Down-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to Zero
3	Counter equal to CMPB on down-count (CBD)
4	Counter equal to CMPA on down-count (CAD)
5 (Lowest)	Counter equal to period (TBPRD)

It is possible to set the compare value greater than the period. In this case the action will take place as shown in [Table 15-20](#).

Table 15-20. Behavior if CMPA/CMPB is Greater than the Period

Counter Mode	Compare on Up-Count Event CAU/CBU	Compare on Down-Count Event CAU/CBU
Up-Count Mode	If $CMPA/CMPB \leq TBPRD$ period, then the event occurs on a compare match ($TBCNT = CMPA$ or $CMPB$). If $CMPA/CMPB > TBPRD$, then the event will not occur.	Never occurs.
Down-Count Mode	Never occurs.	If $CMPA/CMPB < TBPRD$, the event will occur on a compare match ($TBCNT = CMPA$ or $CMPB$). If $CMPA/CMPB \geq TBPRD$, the event will occur on a period match ($TBCNT = TBPRD$).
Up-Down-Count Mode	If $CMPA/CMPB < TBPRD$ and the counter is incrementing, the event occurs on a compare match ($TBCNT = CMPA$ or $CMPB$). If $CMPA/CMPB \geq TBPRD$, the event will occur on a period match ($TBCNT = TBPRD$).	If $CMPA/CMPB < TBPRD$ and the counter is decrementing, the event occurs on a compare match ($TBCNT = CMPA$ or $CMPB$). If $CMPA/CMPB \geq TBPRD$, the event occurs on a period match ($TBCNT = TBPRD$).

15.2.2.5.4 Waveforms for Common Configurations

NOTE: The waveforms in this chapter show the ePWMs behavior for a static compare register value. In a running system, the active compare registers (CMPA and CMPB) are typically updated from their respective shadow registers once every period. The user specifies when the update will take place; either when the time-base counter reaches zero or when the time-base counter reaches period. There are some cases when the action based on the new value can be delayed by one period or the action based on the old value can take effect for an extra period. Some PWM configurations avoid this situation. These include, but are not limited to, the following:

Use up-down-count mode to generate a symmetric PWM:

- If you load CMPA/CMPB on zero, then use CMPA/CMPB values greater than or equal to 1.
- If you load CMPA/CMPB on period, then use CMPA/CMPB values less than or equal to $TBPRD - 1$. This means there will always be a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

Use up-down-count mode to generate an asymmetric PWM:

- To achieve 50%-0% asymmetric PWM use the following configuration: Load CMPA/CMPB on period and use the period action to clear the PWM and a compare-up action to set the PWM. Modulate the compare value from 0 to TBPRD to achieve 50%-0% PWM duty.

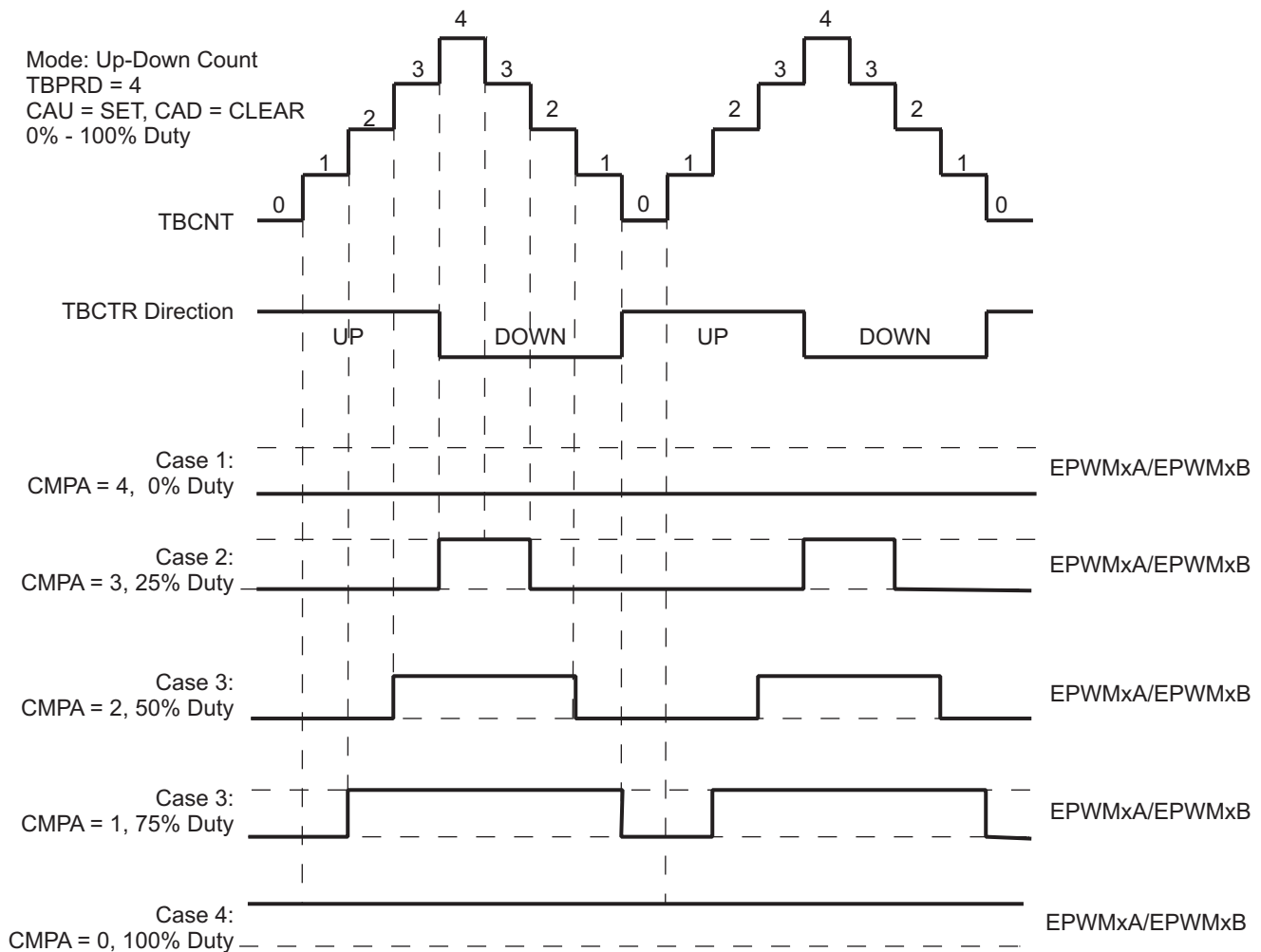
When using up-count mode to generate an asymmetric PWM:

- To achieve 0-100% asymmetric PWM use the following configuration: Load CMPA/CMPB on TBPRD. Use the Zero action to set the PWM and a compare-up action to clear the PWM. Modulate the compare value from 0 to $TBPRD+1$ to achieve 0-100% PWM duty.

Figure 15-27 shows how a symmetric PWM waveform can be generated using the up-down-count mode of the TBCNT. In this mode 0%-100% DC modulation is achieved by using equal compare matches on the up count and down count portions of the waveform. In the example shown, CMPA is used to make the comparison. When the counter is incrementing the CMPA match will pull the PWM output high. Likewise, when the counter is decrementing the compare match will pull the PWM signal low. When CMPA = 0, the PWM signal is low for the entire period giving the 0% duty waveform. When CMPA = TBPRD, the PWM signal is high achieving 100% duty.

When using this configuration in practice, if you load CMPA/CMPB on zero, then use CMPA/CMPB values greater than or equal to 1. If you load CMPA/CMPB on period, then use CMPA/CMPB values less than or equal to TBPRD-1. This means there will always be a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

Figure 15-27. Up-Down-Count Mode Symmetrical Waveform

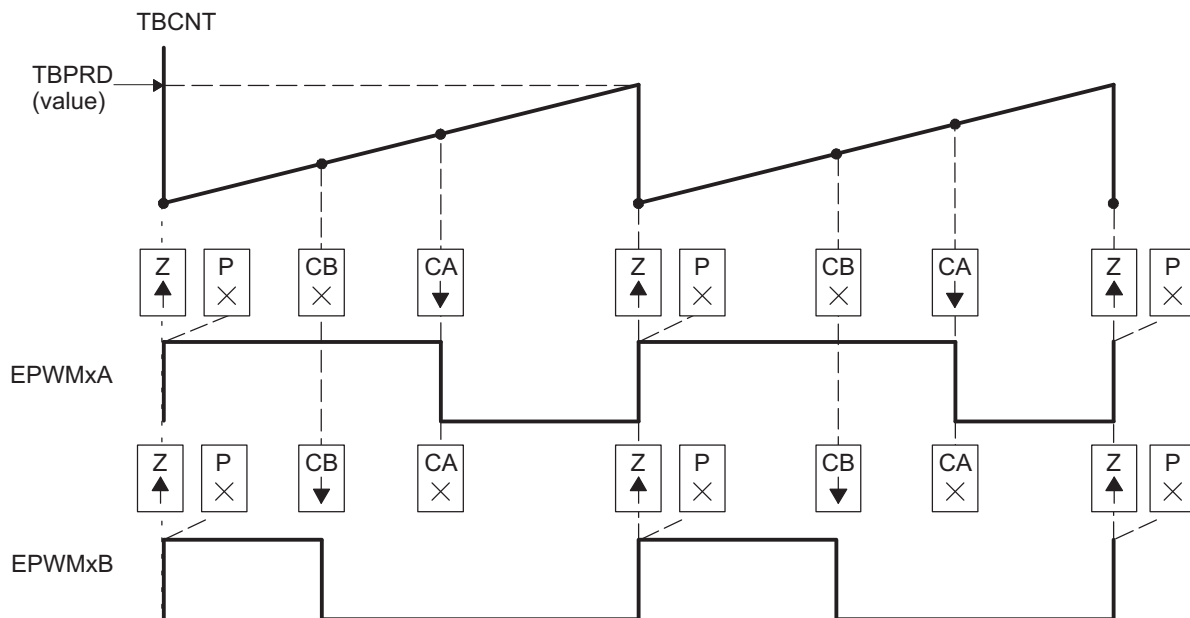


The PWM waveforms in [Figure 15-28](#) through [Figure 15-33](#) show some common action-qualifier configurations. Some conventions used in the figures are as follows:

- TBPRD, CMPA, and CMPB refer to the value written in their respective registers. The active register, not the shadow register, is used by the hardware.
- CMPx, refers to either CMPA or CMPB.
- EPWMxA and EPWMxB refer to the output signals from ePWMx
- Up-Down means Count-up-and-down mode, Up means up-count mode and Dwn means down-count mode
- Sym = Symmetric, Asym = Asymmetric

[Table 15-21](#) and [Table 15-22](#) contains initialization and runtime register configurations for the waveforms in [Figure 15-28](#).

Figure 15-28. Up, Single Edge Asymmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB—Active High



- (1) $\text{PWM period} = (\text{TBPRD} + 1) \times T_{\text{TBCLK}}$
- (2) Duty modulation for EPWMxA is set by CMPA, and is active high (that is, high time duty proportional to CMPA).
- (3) Duty modulation for EPWMxB is set by CMPB and is active high (that is, high time duty proportional to CMPB).
- (4) The "Do Nothing" actions (X) are shown for completeness, but will not be shown on subsequent diagrams.
- (5) Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCNT wraps from period to 0000h.

Table 15-21. EPWMx Initialization for Figure 15-28

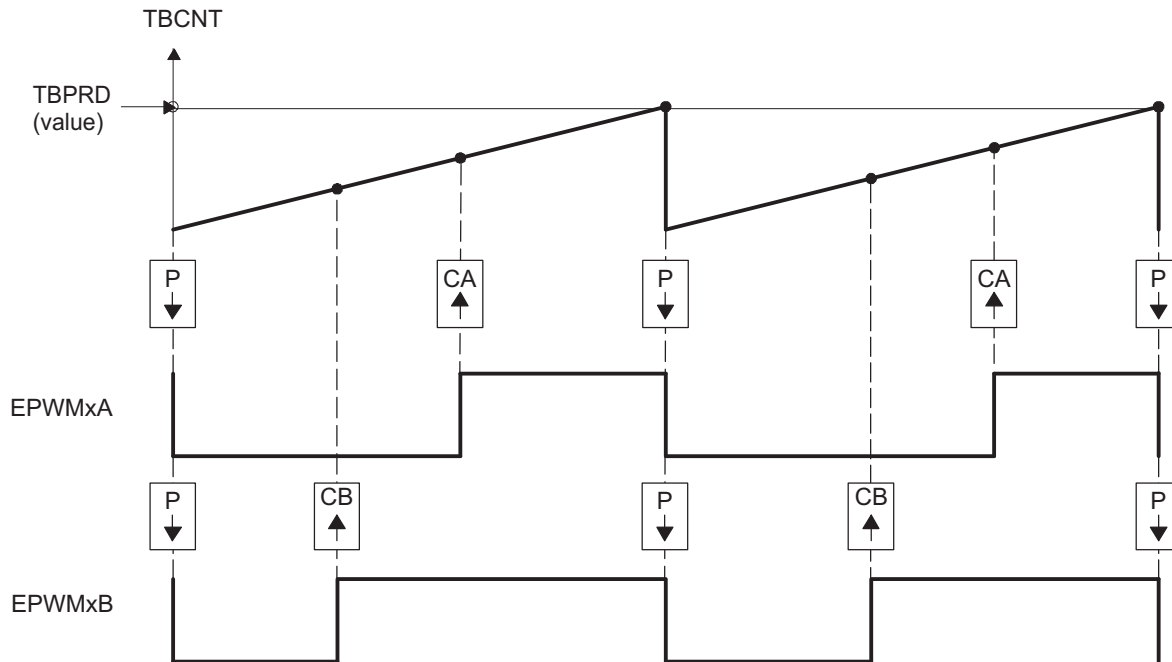
Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	350 (15Eh)	Compare A = 350 TBCLK counts
CMPB	CMPB	200 (C8h)	Compare B = 200 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	ZRO	AQ_SET	
	CAU	AQ_CLEAR	
AQCTLB	ZRO	AQ_SET	
	CBU	AQ_CLEAR	

Table 15-22. EPWMx Run Time Changes for Figure 15-28

Register	Bit	Value	Comments
CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Table 15-23 and Table 15-24 contains initialization and runtime register configurations for the waveforms in Figure 15-29.

Figure 15-29. Up, Single Edge Asymmetric Waveform With Independent Modulation on EPWMxA and EPWMxB—Active Low



- (1) $\text{PWM period} = (\text{TBPRD} + 1) \times T_{\text{TBCLK}}$
- (2) Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).
- (3) Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).
- (4) The Do Nothing actions (X) are shown for completeness here, but will not be shown on subsequent diagrams.
- (5) Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCNT wraps from period to 0000h.

Table 15-23. EPWMx Initialization for Figure 15-29

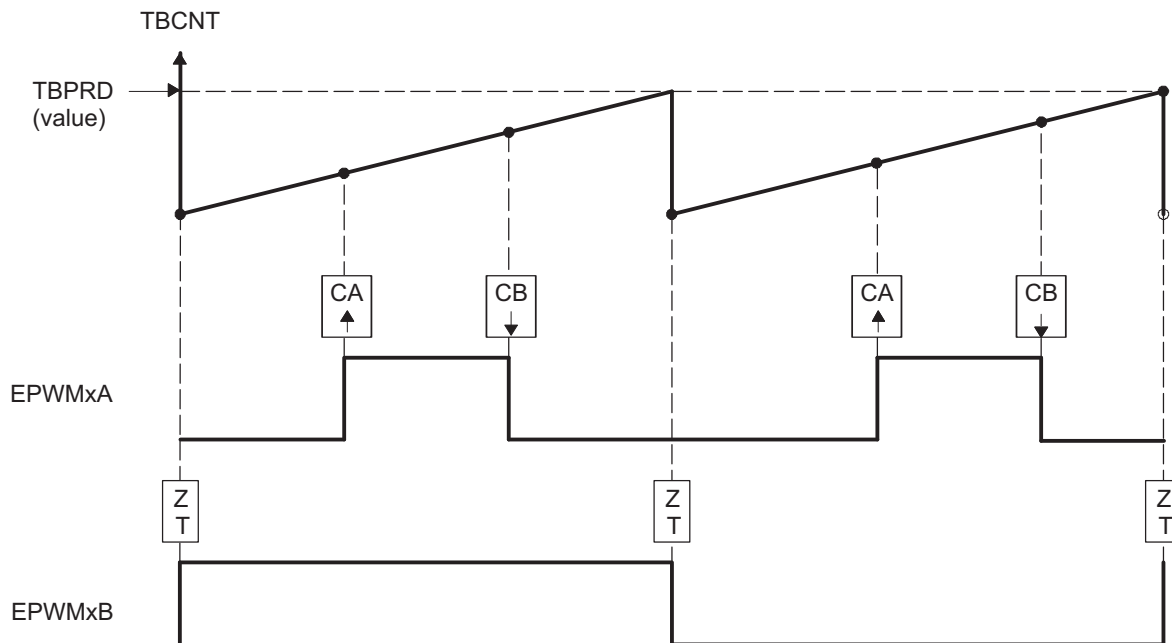
Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	350 (15Eh)	Compare A = 350 TBCLK counts
CMPB	CMPB	200 (C8h)	Compare B = 200 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	PRD	AQ_CLEAR	
	CAU	AQ_SET	
AQCTLB	PRD	AQ_CLEAR	
	CBU	AQ_SET	

Table 15-24. EPWMx Run Time Changes for Figure 15-29

Register	Bit	Value	Comments
CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Table 15-25 and Table 15-26 contains initialization and runtime register configurations for the waveforms Figure 15-30. Use the code in Example 15-1 to define the headers.

Figure 15-30. Up-Count, Pulse Placement Asymmetric Waveform With Independent Modulation on EPWMxA



- (1) $\text{PWM frequency} = 1 / ((\text{TBPRD} + 1) \times T_{\text{TBCLK}})$
- (2) Pulse can be placed anywhere within the PWM cycle (0000h - TBPRD)
- (3) High time duty proportional to (CMPB - CMPA)
- (4) EPWMxB can be used to generate a 50% duty square wave with frequency = $1/2 \times ((\text{TBPRD} + 1) \times \text{TBCLK})$

Table 15-25. EPWMx Initialization for Figure 15-30

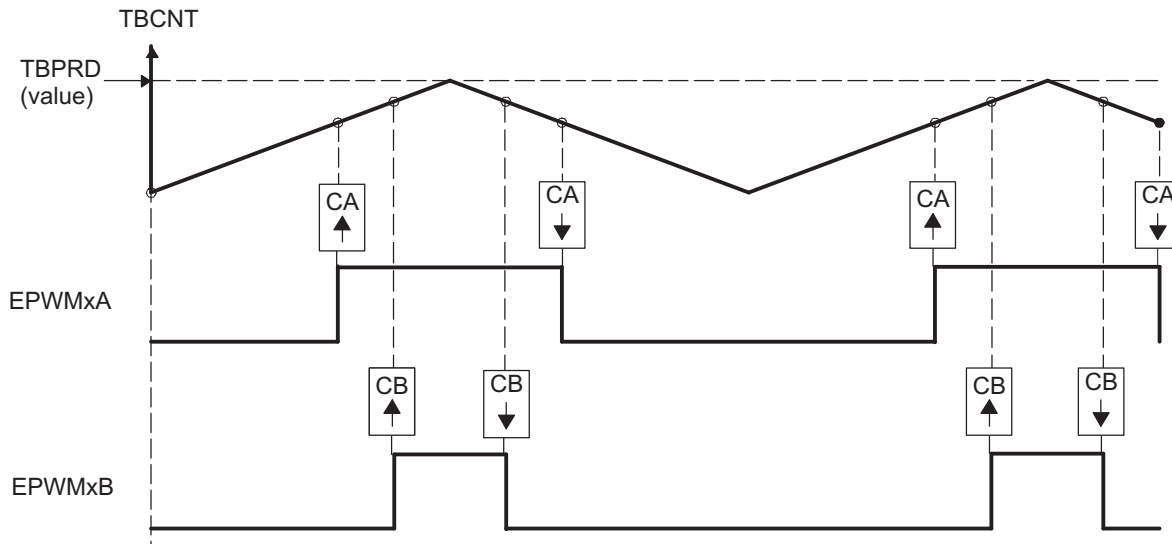
Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	200 (C8h)	Compare A = 200 TBCLK counts
CMPB	CMPB	400 (190h)	Compare B = 400 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	
	CBU	AQ_CLEAR	
AQCTLB	ZRO	AQ_TOGGLE	

Table 15-26. EPWMx Run Time Changes for Figure 15-30

Register	Bit	Value	Comments
CMPA	CMPA	EdgePosA	Adjust duty for output EPWM1A
CMPB	CMPB	EdgePosB	Adjust duty for output EPWM1B

Table 15-27 and Table 15-28 contains initialization and runtime register configurations for the waveforms in Figure 15-31. Use the code in Example 15-1 to define the headers.

Figure 15-31. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Active Low



- (1) $\text{PWM period} = 2 \times \text{TBPRD} \times T_{\text{TBCLK}}$
- (2) Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).
- (3) Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).
- (4) Outputs EPWMxA and EPWMxB can drive independent power switches

Table 15-27. EPWMx Initialization for Figure 15-31

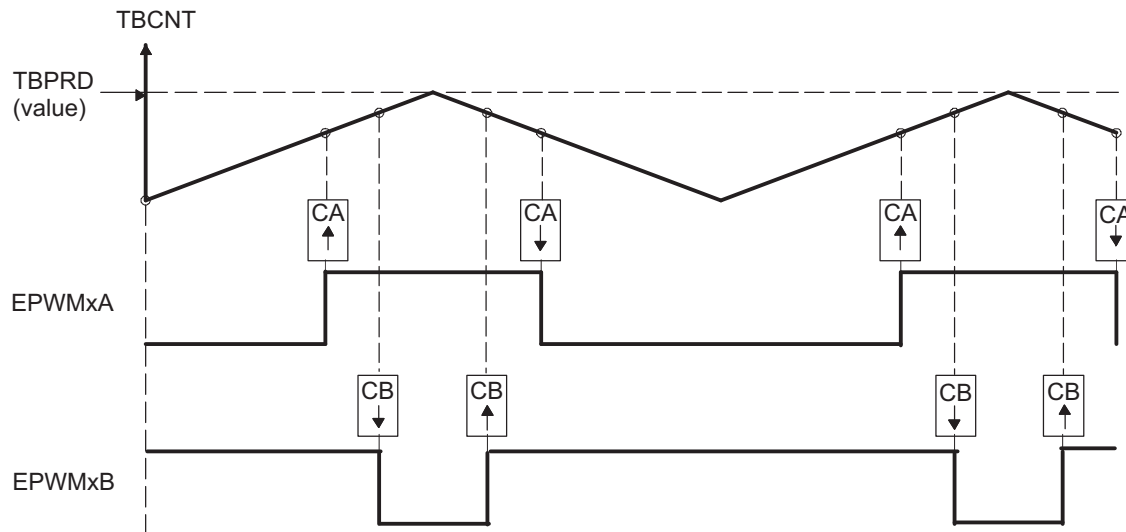
Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	400 (190h)	Compare A = 400 TBCLK counts
CMPB	CMPB	500 (1F4h)	Compare B = 500 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	
	CAD	AQ_CLEAR	
AQCTLB	CBU	AQ_SET	
	CBD	AQ_CLEAR	

Table 15-28. EPWMx Run Time Changes for Figure 15-31

Register	Bit	Value	Comments
CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Table 15-29 and Table 15-30 contains initialization and runtime register configurations for the waveforms in Figure 15-32. Use the code in Example 15-1 to define the headers.

Figure 15-32. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Complementary



- (1) PWM period = $2 \times \text{TBPRD} \times T_{\text{TBCLK}}$
- (2) Duty modulation for EPWMxA is set by CMPA, and is active low, i.e., low time duty proportional to CMPA
- (3) Duty modulation for EPWMxB is set by CMPB and is active high, i.e., high time duty proportional to CMPB
- (4) Outputs EPWMx can drive upper/lower (complementary) power switches
- (5) Dead-band = $\text{CMPB} - \text{CMPA}$ (fully programmable edge placement by software). Note the dead-band module is also available if the more classical edge delay method is required.

Table 15-29. EPWMx Initialization for Figure 15-32

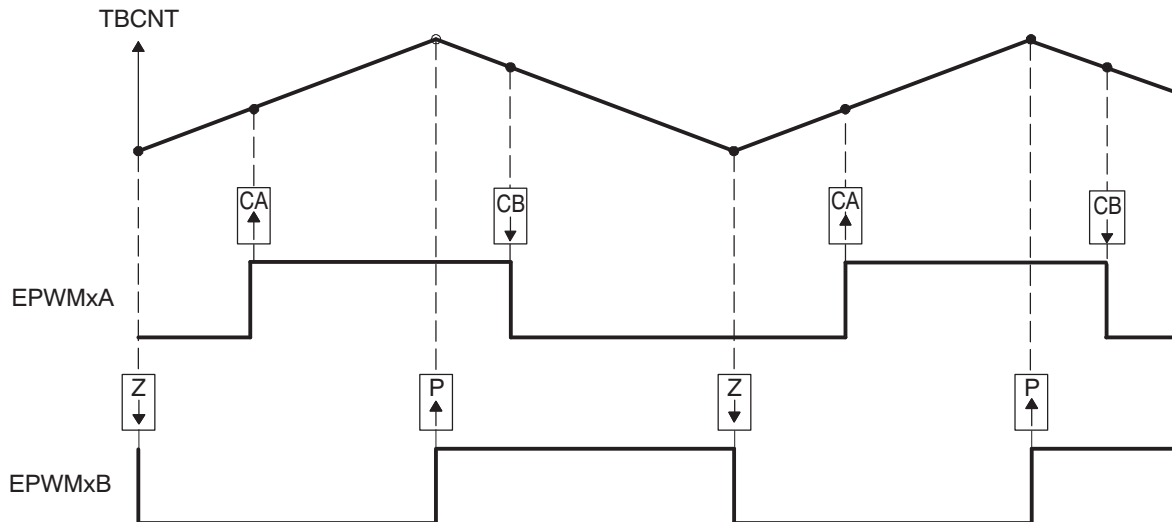
Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	350 (15Eh)	Compare A = 350 TBCLK counts
CMPB	CMPB	400 (190h)	Compare B = 400 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	
	CAD	AQ_CLEAR	
AQCTLB	CBU	AQ_CLEAR	
	CBD	AQ_SET	

Table 15-30. EPWMx Run Time Changes for Figure 15-32

Register	Bit	Value	Comments
CMPA	CMPA	Duty1A	Adjust duty for output EPWM1A
CMPB	CMPB	Duty1B	Adjust duty for output EPWM1B

Table 15-31 and Table 15-32 contains initialization and runtime register configurations for the waveforms in Figure 15-33. Use the code in Example 15-1 to define the headers.

Figure 15-33. Up-Down-Count, Dual Edge Asymmetric Waveform, With Independent Modulation on EPWMxA—Active Low



- (1) PWM period = $2 \times \text{TBPRD} \times \text{TBCLK}$
- (2) Rising edge and falling edge can be asymmetrically positioned within a PWM cycle. This allows for pulse placement techniques.
- (3) Duty modulation for EPWMxA is set by CMPA and CMPB.
- (4) Low time duty for EPWMxA is proportional to (CMPA + CMPB).
- (5) To change this example to active high, CMPA and CMPB actions need to be inverted (i.e., Set ! Clear and Clear Set).
- (6) Duty modulation for EPWMxB is fixed at 50% (utilizes spare action resources for EPWMxB)

Table 15-31. EPWMx Initialization for Figure 15-33

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 601 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCNT	TBCNT	0	Clear TB counter
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
	HSPCLKDIV	TB_DIV1	TBCLK = SYSCLK
	CLKDIV	TB_DIV1	
CMPA	CMPA	250 (FAh)	Compare A = 250 TBCLK counts
CMPB	CMPB	450 (1C2h)	Compare B = 450 TBCLK counts
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	
	CBD	AQ_CLEAR	
AQCTLB	ZRO	AQ_CLEAR	
	PRD	AQ_SET	

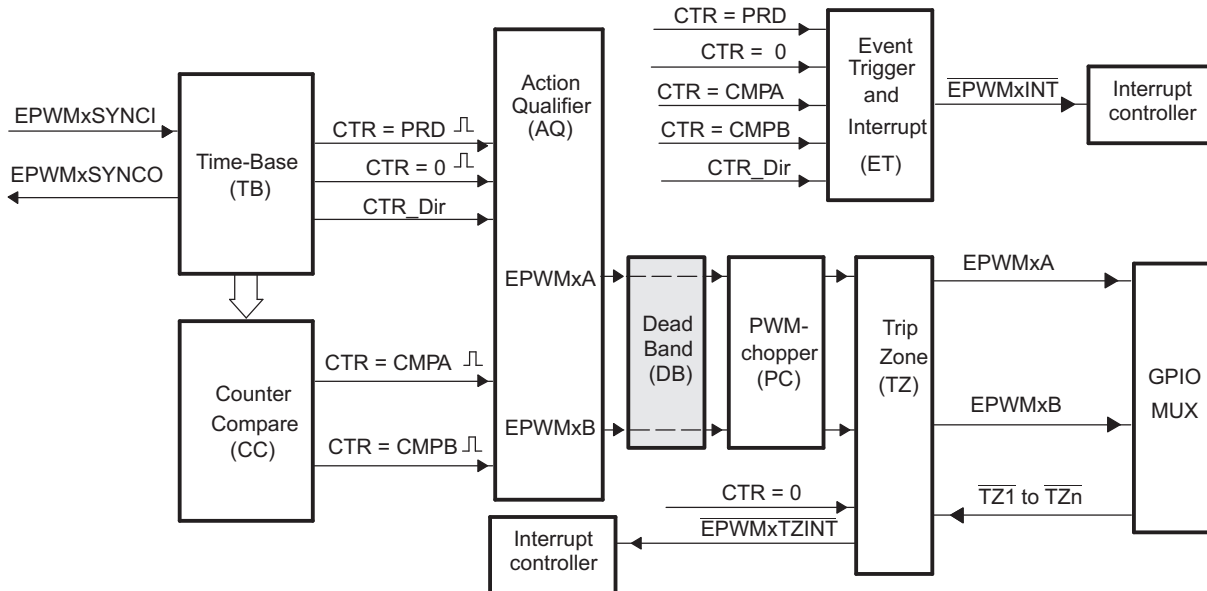
Table 15-32. EPWMx Run Time Changes for Figure 15-33

Register	Bit	Value	Comments
CMPA	CMPA	EdgePosA	Adjust duty for output EPWM1A
CMPB	CMPB	EdgePosB	

15.2.2.6 Dead-Band Generator (DB) Submodule

Figure 15-34 illustrates the dead-band generator submodule within the ePWM module.

Figure 15-34. Dead-Band Generator Submodule



15.2.2.6.1 Purpose of the Dead-Band Submodule

The "Action-qualifier (AQ) Module" section discussed how it is possible to generate the required dead-band by having full control over edge placement using both the CMPA and CMPB resources of the ePWM module. However, if the more classical edge delay-based dead-band with polarity control is required, then the dead-band generator submodule should be used.

The key functions of the dead-band generator submodule are:

- Generating appropriate signal pairs (EPWMxA and EPWMxB) with dead-band relationship from a single EPWMxA input
- Programming signal pairs for:
 - Active high (AH)
 - Active low (AL)
 - Active high complementary (AHC)
 - Active low complementary (ALC)
- Adding programmable delay to rising edges (RED)
- Adding programmable delay to falling edges (FED)
- Can be totally bypassed from the signal path (note dotted lines in diagram)

15.2.2.6.2 Controlling and Monitoring the Dead-Band Submodule

The dead-band generator submodule operation is controlled and monitored via the following registers:

Table 15-33. Dead-Band Generator Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
DBCTL	Dead-Band Control Register	1Eh	No
DBRED	Dead-Band Rising Edge Delay Count Register	20h	No
DBFED	Dead-Band Falling Edge Delay Count Register	22h	No

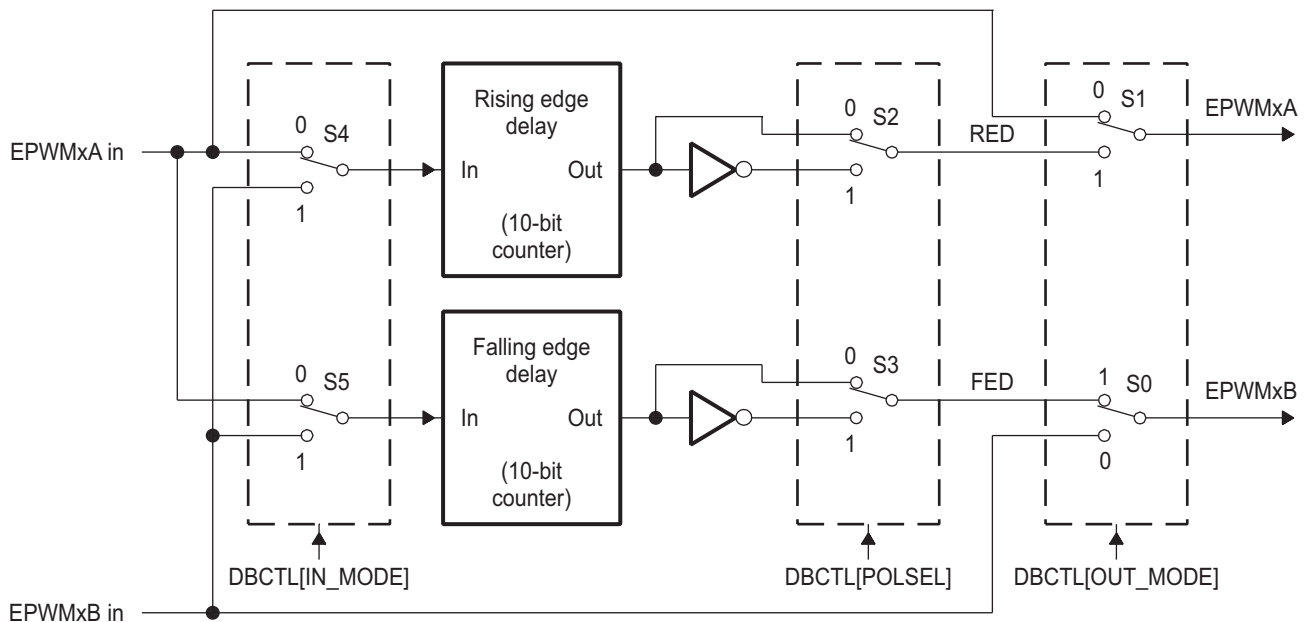
15.2.2.6.3 Operational Highlights for the Dead-Band Generator Submodule

The following sections provide the operational highlights.

The dead-band submodule has two groups of independent selection options as shown in Figure 15-35.

- Input Source Selection:** The input signals to the dead-band module are the EPWMxA and EPWMxB output signals from the action-qualifier. In this section they will be referred to as EPWMxA In and EPWMxB In. Using the DBCTL[IN_MODE] control bits, the signal source for each delay, falling-edge or rising-edge, can be selected:
 - EPWMxA In is the source for both falling-edge and rising-edge delay. This is the default mode.
 - EPWMxA In is the source for falling-edge delay, EPWMxB In is the source for rising-edge delay.
 - EPWMxA In is the source for rising edge delay, EPWMxB In is the source for falling-edge delay.
 - EPWMxB In is the source for both falling-edge and rising-edge delay.
- Output Mode Control:** The output mode is configured by way of the DBCTL[OUT_MODE] bits. These bits determine if the falling-edge delay, rising-edge delay, neither, or both are applied to the input signals.
- Polarity Control:** The polarity control (DBCTL[POLSEL]) allows you to specify whether the rising-edge delayed signal and/or the falling-edge delayed signal is to be inverted before being sent out of the dead-band submodule.

Figure 15-35. Configuration Options for the Dead-Band Generator Submodule



Although all combinations are supported, not all are typical usage modes. [Table 15-34](#) lists some classical dead-band configurations. These modes assume that the DBCTL[IN_MODE] is configured such that EPWMxA In is the source for both falling-edge and rising-edge delay. Enhanced, or non-traditional modes can be achieved by changing the input signal source. The modes shown in [Table 15-34](#) fall into the following categories:

- **Mode 1: Bypass both falling-edge delay (FED) and rising-edge delay (RED)** Allows you to fully disable the dead-band submodule from the PWM signal path.
- **Mode 2-5: Classical Dead-Band Polarity Settings** These represent typical polarity configurations that should address all the active high/low modes required by available industry power switch gate drivers. The waveforms for these typical cases are shown in [Figure 15-36](#). Note that to generate equivalent waveforms to [Figure 15-36](#), configure the action-qualifier submodule to generate the signal as shown for EPWMxA.
- **Mode 6: Bypass rising-edge-delay and Mode 7: Bypass falling-edge-delay** Finally the last two entries in [Table 15-34](#) show combinations where either the falling-edge-delay (FED) or rising-edge-delay (RED) blocks are bypassed.

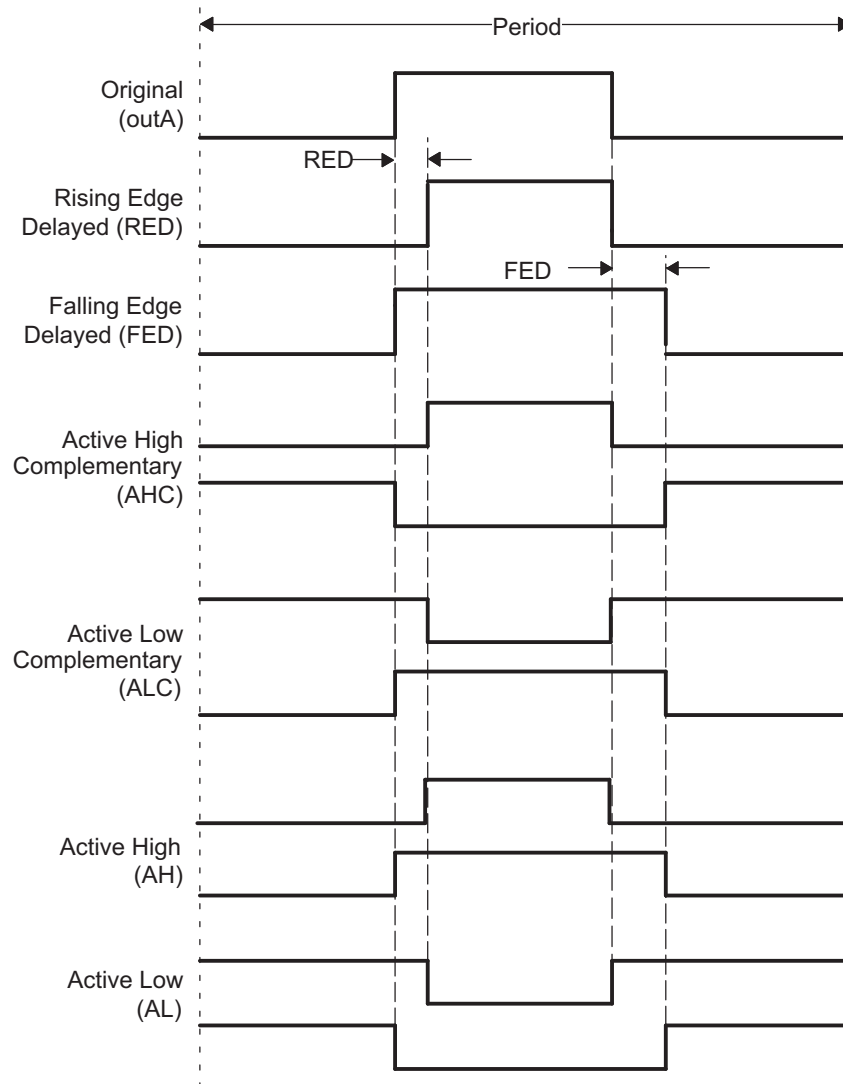
Table 15-34. Classical Dead-Band Operating Modes

Mode	Mode Description ⁽¹⁾	DBCTL[POLSEL]		DBCTL[OUT_MODE]	
		S3	S2	S1	S0
1	EPWMxA and EPWMxB Passed Through (No Delay)	x	x	0	0
2	Active High Complementary (AHC)	1	0	1	1
3	Active Low Complementary (ALC)	0	1	1	1
4	Active High (AH)	0	0	1	1
5	Active Low (AL)	1	1	1	1
6	EPWMxA Out = EPWMxA In (No Delay) EPWMxB Out = EPWMxA In with Falling Edge Delay	0 or 1	0 or 1	0	1
7	EPWMxA Out = EPWMxA In with Rising Edge Delay EPWMxB Out = EPWMxB In with No Delay	0 or 1	0 or 1	1	0

⁽¹⁾ These are classical dead-band modes and assume that DBCTL[IN_MODE] = 0,0. That is, EPWMxA in is the source for both the falling-edge and rising-edge delays. Enhanced, non-traditional modes can be achieved by changing the IN_MODE configuration.

Figure 15-36 shows waveforms for typical cases where 0% < duty < 100%.

Figure 15-36. Dead-Band Waveforms for Typical Cases (0% < Duty < 100%)



The dead-band submodule supports independent values for rising-edge (RED) and falling-edge (FED) delays. The amount of delay is programmed using the DBRED and DBFED registers. These are 10-bit registers and their value represents the number of time-base clock, TBCLK, periods a signal edge is delayed by. For example, the formula to calculate falling-edge-delay and rising-edge-delay are:

$$FED = DBFED \times T_{TBCLK}$$

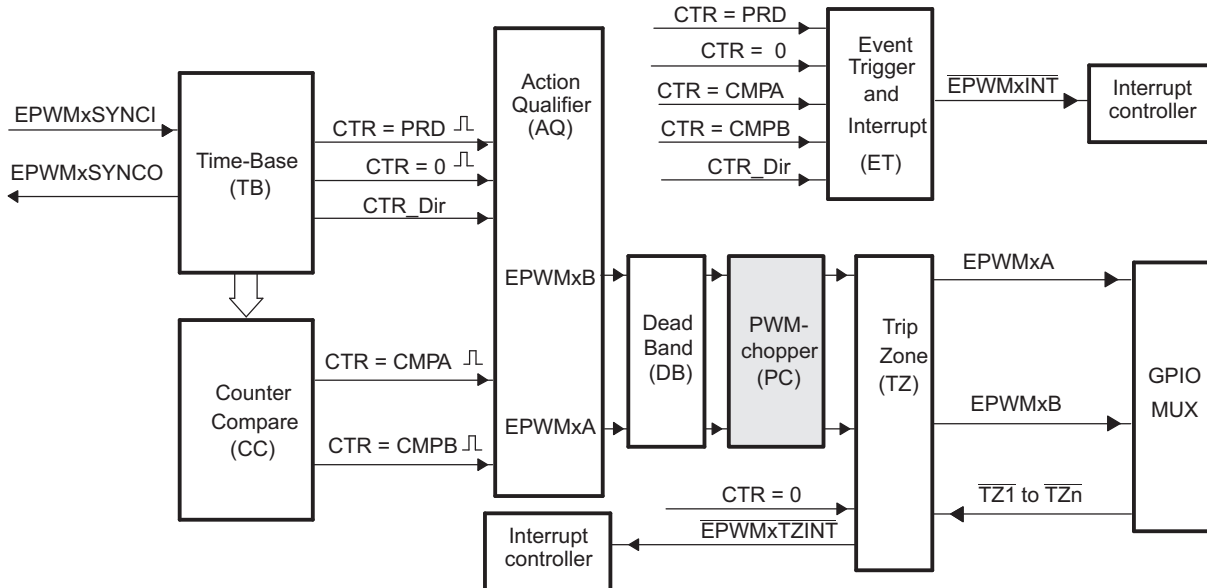
$$RED = DBRED \times T_{TBCLK}$$

Where T_{TBCLK} is the period of TBCLK, the prescaled version of SYSCLKOUT.

15.2.2.7 PWM-Chopper (PC) Submodule

Figure 15-37 illustrates the PWM-chopper (PC) submodule within the ePWM module. The PWM-chopper submodule allows a high-frequency carrier signal to modulate the PWM waveform generated by the action-qualifier and dead-band submodules. This capability is important if you need pulse transformer-based gate drivers to control the power switching elements.

Figure 15-37. PWM-Chopper Submodule



15.2.2.7.1 Purpose of the PWM-Chopper Submodule

The key functions of the PWM-chopper submodule are:

- Programmable chopping (carrier) frequency
- Programmable pulse width of first pulse
- Programmable duty cycle of second and subsequent pulses
- Can be fully bypassed if not required

15.2.2.7.2 Controlling the PWM-Chopper Submodule

The PWM-chopper submodule operation is controlled via the register in [Table 15-35](#).

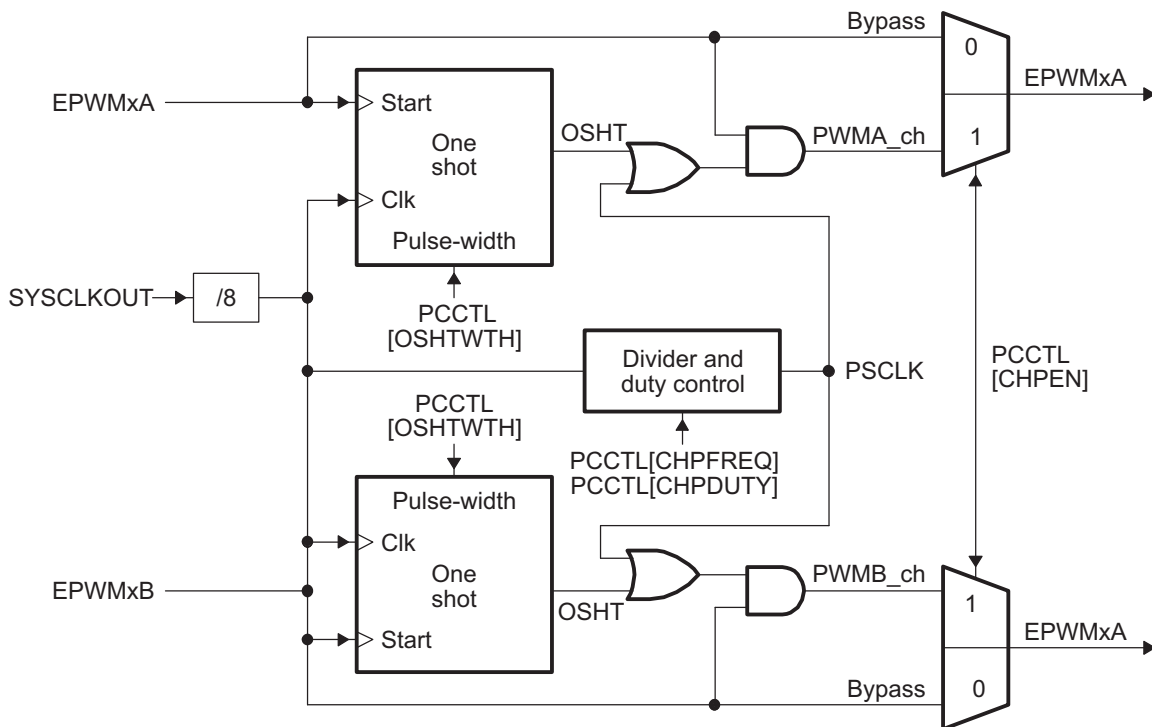
Table 15-35. PWM-Chopper Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
PCCTL	PWM-chopper Control Register	3Ch	No

15.2.2.7.3 Operational Highlights for the PWM-Chopper Submodule

Figure 15-38 shows the operational details of the PWM-chopper submodule. The carrier clock is derived from SYSCLKOUT. Its frequency and duty cycle are controlled via the CHPFREQ and CHPDUTY bits in the PCCTL register. The one-shot block is a feature that provides a high energy first pulse to ensure hard and fast power switch turn on, while the subsequent pulses sustain pulses, ensuring the power switch remains on. The one-shot width is programmed via the OSHTWTH bits. The PWM-chopper submodule can be fully disabled (bypassed) via the CHPEN bit.

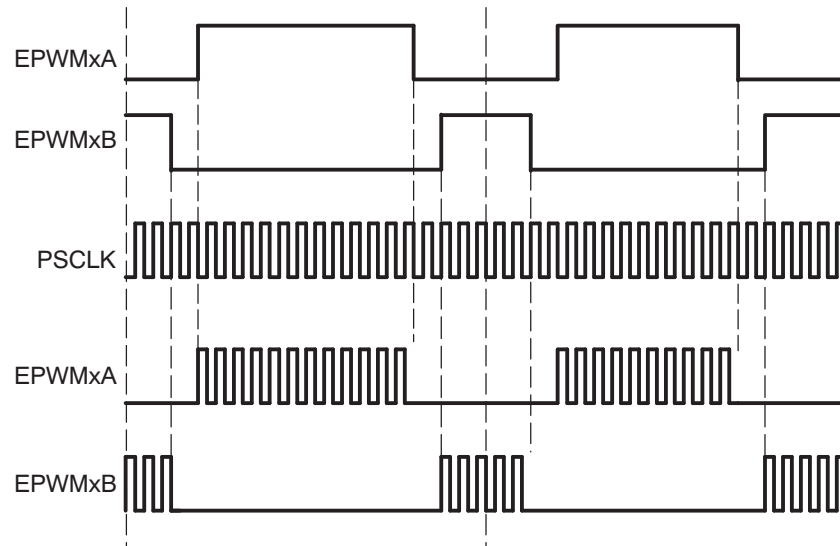
Figure 15-38. PWM-Chopper Submodule Signals and Registers



15.2.2.7.4 Waveforms

Figure 15-39 shows simplified waveforms of the chopping action only; one-shot and duty-cycle control are not shown. Details of the one-shot and duty-cycle control are discussed in the following sections.

Figure 15-39. Simple PWM-Chopper Submodule Waveforms Showing Chopping Action Only



15.2.2.7.4.1 One-Shot Pulse

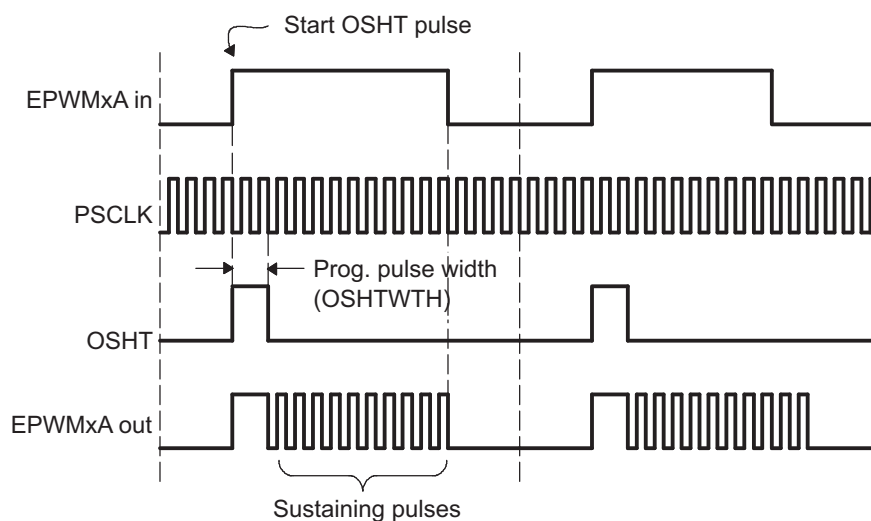
The width of the first pulse can be programmed to any of 16 possible pulse width values. The width or period of the first pulse is given by:

$$T_{1\text{stpulse}} = T_{\text{SYSCLKOUT}} \times 8 \times \text{OSHTWTH}$$

Where $T_{\text{SYSCLKOUT}}$ is the period of the system clock (SYSCLKOUT) and OSHTWTH is the four control bits (value from 1 to 16)

Figure 15-40 shows the first and subsequent sustaining pulses.

Figure 15-40. PWM-Chopper Submodule Waveforms Showing the First Pulse and Subsequent Sustaining Pulses

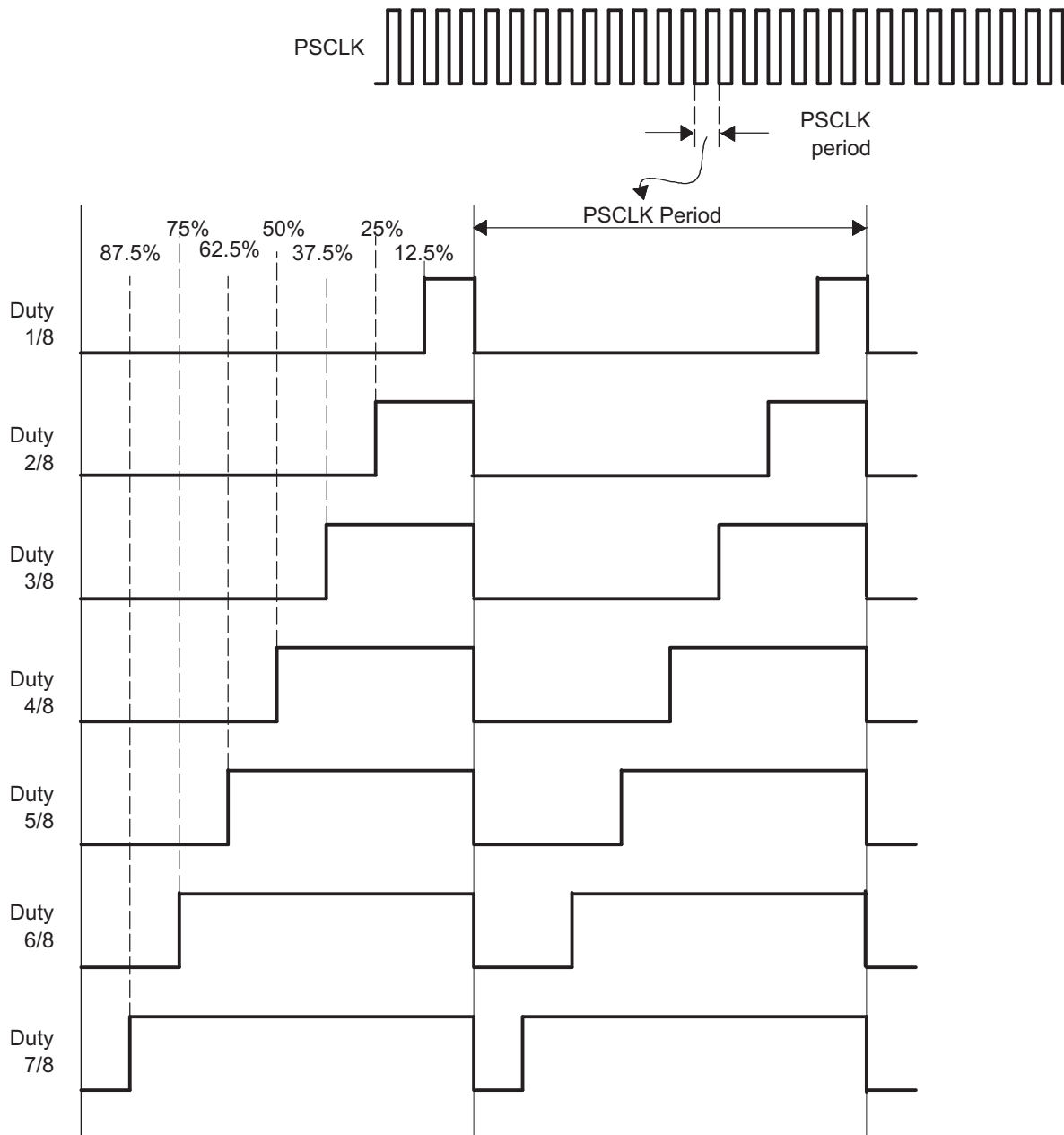


15.2.2.7.4.2 Duty Cycle Control

Pulse transformer-based gate drive designs need to comprehend the magnetic properties or characteristics of the transformer and associated circuitry. Saturation is one such consideration. To assist the gate drive designer, the duty cycles of the second and subsequent pulses have been made programmable. These sustaining pulses ensure the correct drive strength and polarity is maintained on the power switch gate during the on period, and hence a programmable duty cycle allows a design to be tuned or optimized via software control.

Figure 15-41 shows the duty cycle control that is possible by programming the CHPDUTY bits. One of seven possible duty ratios can be selected ranging from 12.5% to 87.5%.

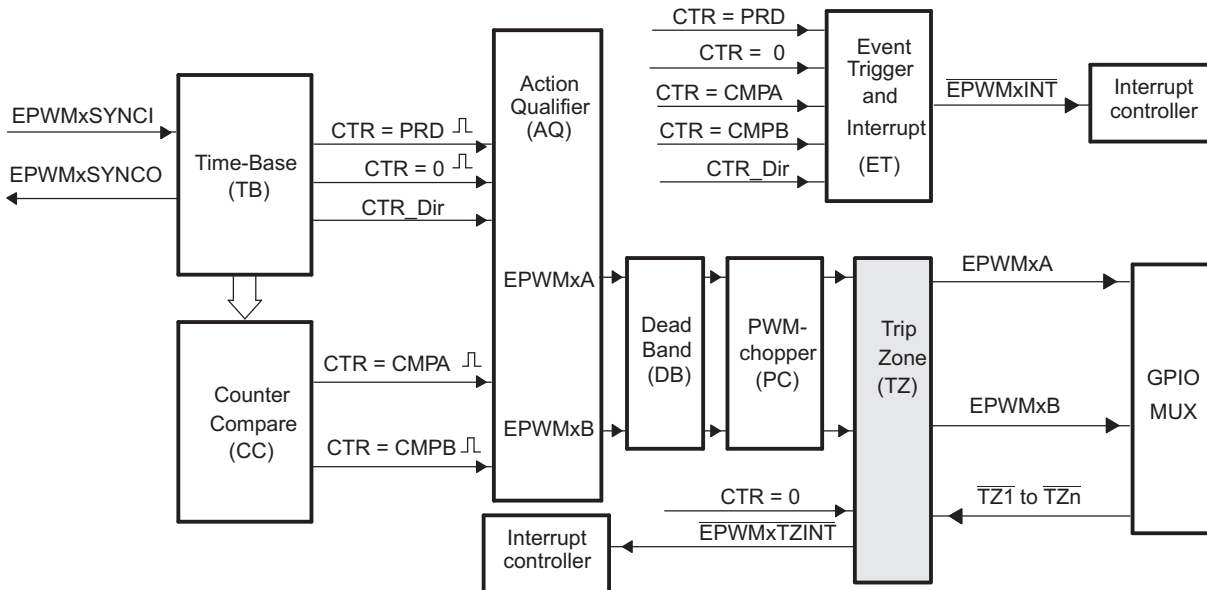
Figure 15-41. PWM-Chopper Submodule Waveforms Showing the Pulse Width (Duty Cycle) Control of Sustaining Pulses



15.2.2.8 Trip-Zone (TZ) Submodule

Figure 15-42 shows how the trip-zone (TZ) submodule fits within the ePWM module. Each ePWM module is connected to every \overline{TZ} signal that are sourced from the GPIO MUX. These signals indicates external fault or trip conditions, and the ePWM outputs can be programmed to respond accordingly when faults occur. See Section 15.1.2 to determine the number of trip-zone pins available for the device.

Figure 15-42. Trip-Zone Submodule



15.2.2.8.1 Purpose of the Trip-Zone Submodule

The key functions of the trip-zone submodule are:

- Trip inputs $\overline{TZ1}$ to \overline{TZn} can be flexibly mapped to any ePWM module.
- Upon a fault condition, outputs $EPWMxA$ and $EPWMxB$ can be forced to one of the following:
 - High
 - Low
 - High-impedance
 - No action taken
- Support for one-shot trip (OSHT) for major short circuits or over-current conditions.
- Support for cycle-by-cycle tripping (CBC) for current limiting operation.
- Each trip-zone input pin can be allocated to either one-shot or cycle-by-cycle operation.
- Interrupt generation is possible on any trip-zone pin.
- Software-forced tripping is also supported.
- The trip-zone submodule can be fully bypassed if it is not required.

15.2.2.8.2 Controlling and Monitoring the Trip-Zone Submodule

The trip-zone submodule operation is controlled and monitored through the following registers:

Table 15-36. Trip-Zone Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
TZSEL	Trip-Zone Select Register	24h	No
TZCTL	Trip-Zone Control Register	28h	No
TZEINT	Trip-Zone Enable Interrupt Register	2Ah	No
TZFLG	Trip-Zone Flag Register	2Ch	No
TZCLR	Trip-Zone Clear Register	2Eh	No
TZFRC	Trip-Zone Force Register	30h	No

15.2.2.8.3 Operational Highlights for the Trip-Zone Submodule

The following sections describe the operational highlights and configuration options for the trip-zone submodule.

The trip-zone signals at pin $\overline{TZ1}$ to \overline{TZn} is an active-low input signal. When the pin goes low, it indicates that a trip event has occurred. Each ePWM module can be individually configured to ignore or use each of the trip-zone pins. Which trip-zone pins are used by a particular ePWM module is determined by the TZSEL register for that specific ePWM module. The trip-zone signal may or may not be synchronized to the system clock (SYSCLKOUT). A minimum of 1 SYSCLKOUT low pulse on the \overline{TZn} inputs is sufficient to trigger a fault condition in the ePWM module. The asynchronous trip makes sure that if clocks are missing for any reason, the outputs can still be tripped by a valid event present on the \overline{TZn} inputs.

The \overline{TZn} input can be individually configured to provide either a cycle-by-cycle or one-shot trip event for a ePWM module. The configuration is determined by the TZSEL[CBCn] and TZSEL[OSHTn] bits (where n corresponds to the trip pin) respectively.

- Cycle-by-Cycle (CBC):** When a cycle-by-cycle trip event occurs, the action specified in the TZCTL register is carried out immediately on the EPWMxA and/or EPWMxB output. [Table 15-37](#) lists the possible actions. In addition, the cycle-by-cycle trip event flag (TZFLG[CBC]) is set and a EPWMxTZINT interrupt is generated if it is enabled in the TZEINT register.

The specified condition on the pins is automatically cleared when the ePWM time-base counter reaches zero (TBCNT = 0000h) if the trip event is no longer present. Therefore, in this mode, the trip event is cleared or reset every PWM cycle. The TZFLG[CBC] flag bit will remain set until it is manually cleared by writing to the TZCLR[CBC] bit. If the cycle-by-cycle trip event is still present when the TZFLG[CBC] bit is cleared, then it will again be immediately set.
- One-Shot (OSHT):** When a one-shot trip event occurs, the action specified in the TZCTL register is carried out immediately on the EPWMxA and/or EPWMxB output. [Table 15-37](#) lists the possible actions. In addition, the one-shot trip event flag (TZFLG[OST]) is set and a EPWMxTZINT interrupt is generated if it is enabled in the TZEINT register. The one-shot trip condition must be cleared manually by writing to the TZCLR[OST] bit.

The action taken when a trip event occurs can be configured individually for each of the ePWM output pins by way of the TZCTL[TZA] and TZCTL[TZB] register bits. One of four possible actions, shown in [Table 15-37](#), can be taken on a trip event.

Table 15-37. Possible Actions On a Trip Event

TZCTL[TZA] and/or TZCTL[TZB]	EPWMxA and/or EPWMxB	Comment
0	High-Impedance	Tripped
1h	Force to High State	Tripped
2h	Force to Low State	Tripped
3h	No Change	Do Nothing. No change is made to the output.

Example 15-2. Trip-Zone Configurations
Scenario A:

A one-shot trip event on $\overline{TZ1}$ pulls both EPWM1A, EPWM1B low and also forces EPWM2A and EPWM2B high.

- Configure the ePWM1 registers as follows:
 - TZSEL[OSHT1] = 1: enables \overline{TZ} as a one-shot event source for ePWM1
 - TZCTL[TZA] = 2: EPWM1A will be forced low on a trip event.
 - TZCTL[TZB] = 2: EPWM1B will be forced low on a trip event.
- Configure the ePWM2 registers as follows:
 - TZSEL[OSHT1] = 1: enables \overline{TZ} as a one-shot event source for ePWM2
 - TZCTL[TZA] = 1: EPWM2A will be forced high on a trip event.
 - TZCTL[TZB] = 1: EPWM2B will be forced high on a trip event.

Scenario B:

A cycle-by-cycle event on $\overline{TZ5}$ pulls both EPWM1A, EPWM1B low.

A one-shot event on $\overline{TZ1}$ or $\overline{TZ6}$ puts EPWM2A into a high impedance state.

- Configure the ePWM1 registers as follows:
 - TZSEL[CBC5] = 1: enables $\overline{TZ5}$ as a one-shot event source for ePWM1
 - TZCTL[TZA] = 2: EPWM1A will be forced low on a trip event.
 - TZCTL[TZB] = 2: EPWM1B will be forced low on a trip event.
- Configure the ePWM2 registers as follows:
 - TZSEL[OSHT1] = 1: enables $\overline{TZ1}$ as a one-shot event source for ePWM2
 - TZSEL[OSHT6] = 1: enables $\overline{TZ6}$ as a one-shot event source for ePWM1
 - TZCTL[TZA] = 0: EPWM1A will be put into a high-impedance state on a trip event.
 - TZCTL[TZB] = 3: EPWM1B will ignore the trip event.

15.2.2.8.4 Generating Trip Event Interrupts

Figure 15-43 and Figure 15-44 illustrate the trip-zone submodule control and interrupt logic, respectively.

Figure 15-43. Trip-Zone Submodule Mode Control Logic

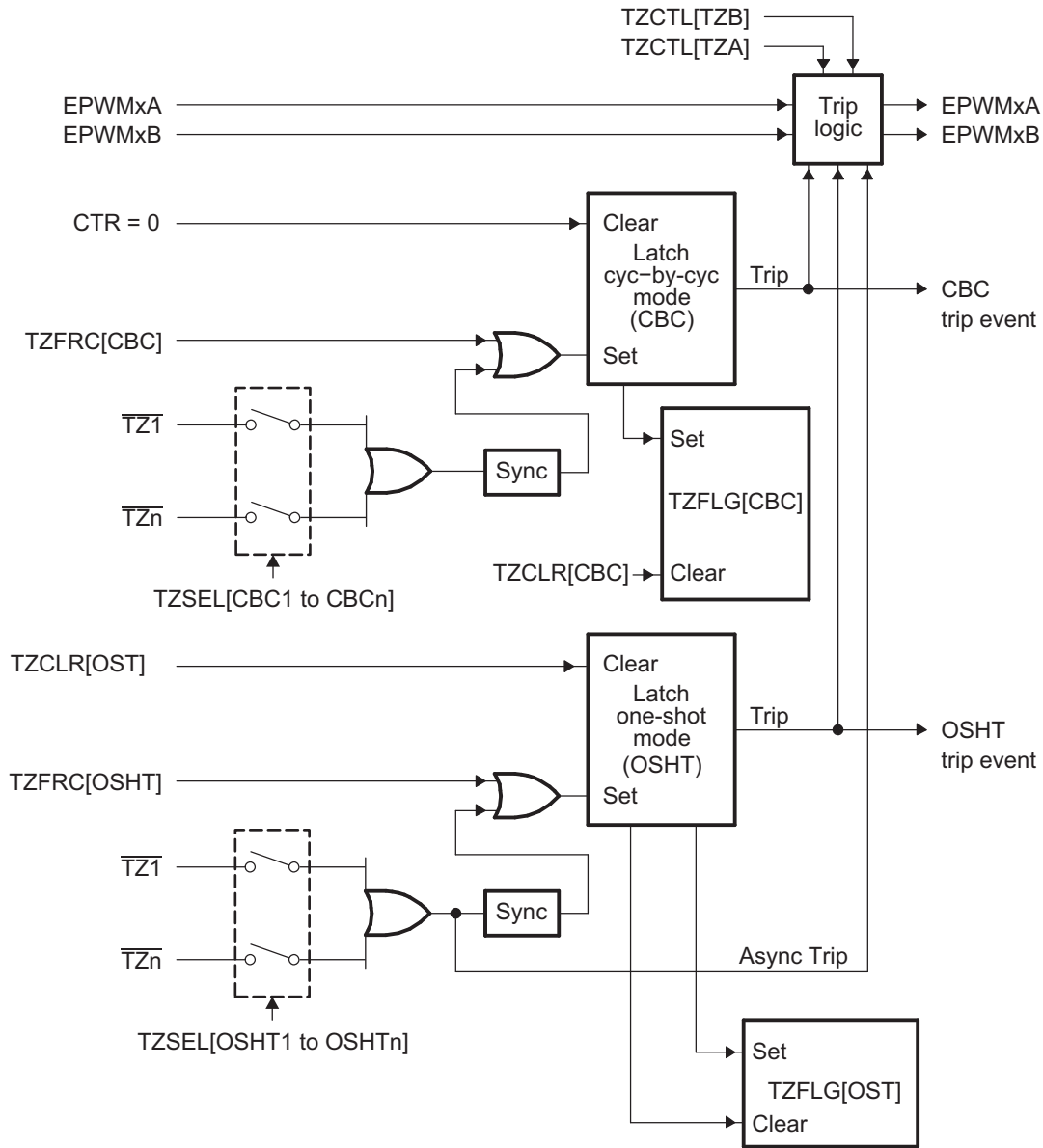
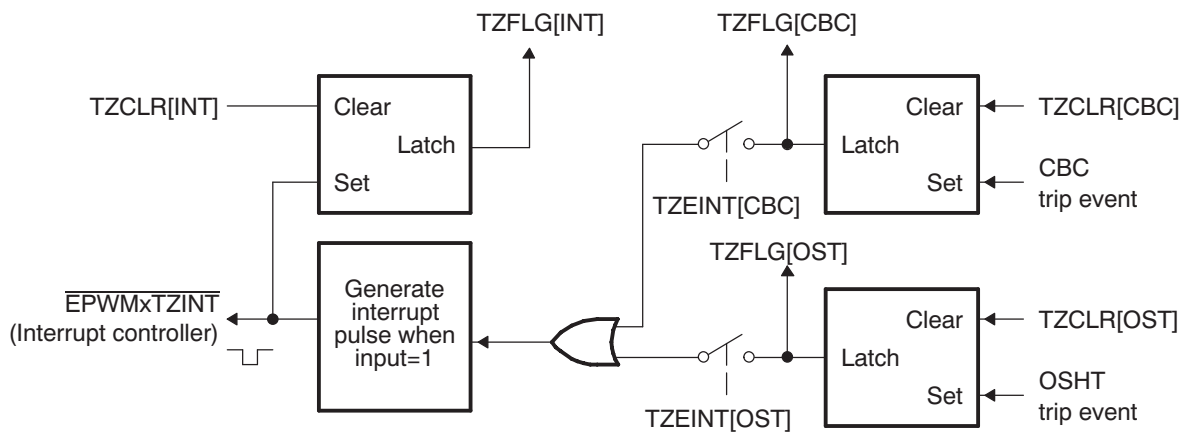


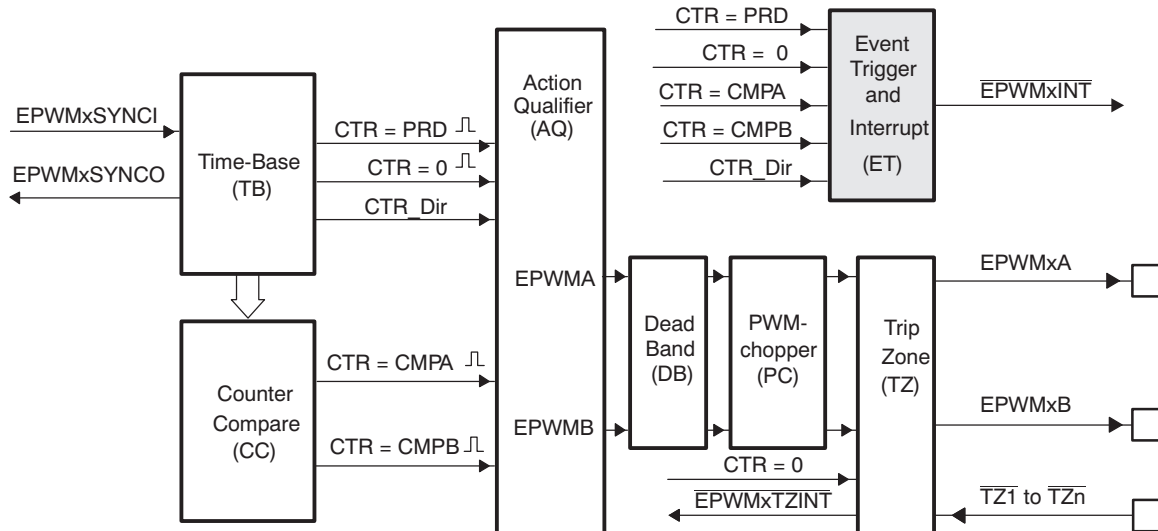
Figure 15-44. Trip-Zone Submodule Interrupt Logic



15.2.2.9 Event-Trigger (ET) Submodule

Figure 15-45 shows the event-trigger (ET) submodule in the ePWM system. The event-trigger submodule manages the events generated by the time-base submodule and the counter-compare submodule to generate an interrupt to the CPU.

Figure 15-45. Event-Trigger Submodule



15.2.2.9.1 Purpose of the Event-Trigger Submodule

The key functions of the event-trigger submodule are:

- Receives event inputs generated by the time-base and counter-compare submodules
- Uses the time-base direction information for up/down event qualification
- Uses prescaling logic to issue interrupt requests at:
 - Every event
 - Every second event
 - Every third event
- Provides full visibility of event generation via event counters and flags

15.2.2.9.2 Controlling and Monitoring the Event-Trigger Submodule

The key registers used to configure the event-trigger submodule are shown in [Table 15-38](#):

Table 15-38. Event-Trigger Submodule Registers

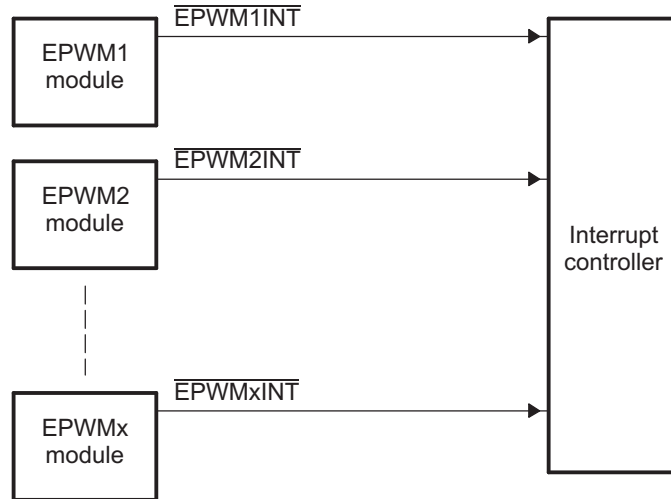
Acronym	Register Description	Address Offset	Shadowed
ETSEL	Event-Trigger Selection Register	32h	No
ETPS	Event-Trigger Prescale Register	34h	No
ETFLG	Event-Trigger Flag Register	36h	No
ETCLR	Event-Trigger Clear Register	38h	No
ETFRC	Event-Trigger Force Register	3Ah	No

15.2.2.9.3 Operational Overview of the Event-Trigger Submodule

The following sections describe the event-trigger submodule's operational highlights.

Each ePWM module has one interrupt request line connected to the interrupt controller as shown in Figure 15-46.

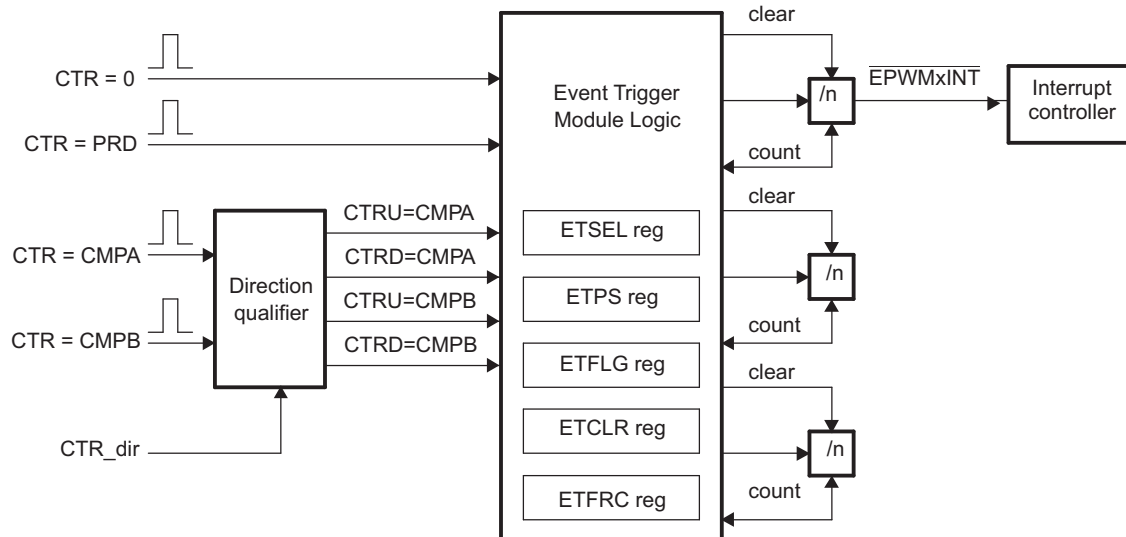
Figure 15-46. Event-Trigger Submodule Inter-Connectivity to Interrupt Controller



The event-trigger submodule monitors various event conditions (the left side inputs to event-trigger submodule shown in Figure 15-47) and can be configured to prescale these events before issuing an Interrupt request. The event-trigger prescaling logic can issue Interrupt requests at:

- Every event
- Every second event
- Every third event

Figure 15-47. Event-Trigger Submodule Showing Event Inputs and Prescaled Outputs



- ETSEL—This selects which of the possible events will trigger an interrupt.
- ETPS—This programs the event prescaling options previously mentioned.
- ETFLG—These are flag bits indicating status of the selected and prescaled events.
- ETCLR—These bits allow you to clear the flag bits in the ETFLG register via software.
- ETFRC—These bits allow software forcing of an event. Useful for debugging or software intervention.

A more detailed look at how the various register bits interact with the Interrupt is shown in [Figure 15-48](#).

[Figure 15-48](#) shows the event-trigger's interrupt generation logic. The interrupt-period (ETPS[INTPRD]) bits specify the number of events required to cause an interrupt pulse to be generated. The choices available are:

- Do not generate an interrupt
- Generate an interrupt on every event
- Generate an interrupt on every second event
- Generate an interrupt on every third event

An interrupt cannot be generated on every fourth or more events.

Which event can cause an interrupt is configured by the interrupt selection (ETSEL[INTSEL]) bits. The event can be one of the following:

- Time-base counter equal to zero (TBCNT = 0000h).
- Time-base counter equal to period (TBCNT = TBPRD).
- Time-base counter equal to the compare A register (CMPA) when the timer is incrementing.
- Time-base counter equal to the compare A register (CMPA) when the timer is decrementing.
- Time-base counter equal to the compare B register (CMPB) when the timer is incrementing.
- Time-base counter equal to the compare B register (CMPB) when the timer is decrementing.

The number of events that have occurred can be read from the interrupt event counter (ETPS[INTCNT]) register bits. That is, when the specified event occurs the ETPS[INTCNT] bits are incremented until they reach the value specified by ETPS[INTPRD]. When ETPS[INTCNT] = ETPS[INTPRD] the counter stops counting and its output is set. The counter is only cleared when an interrupt is sent to the interrupt controller.

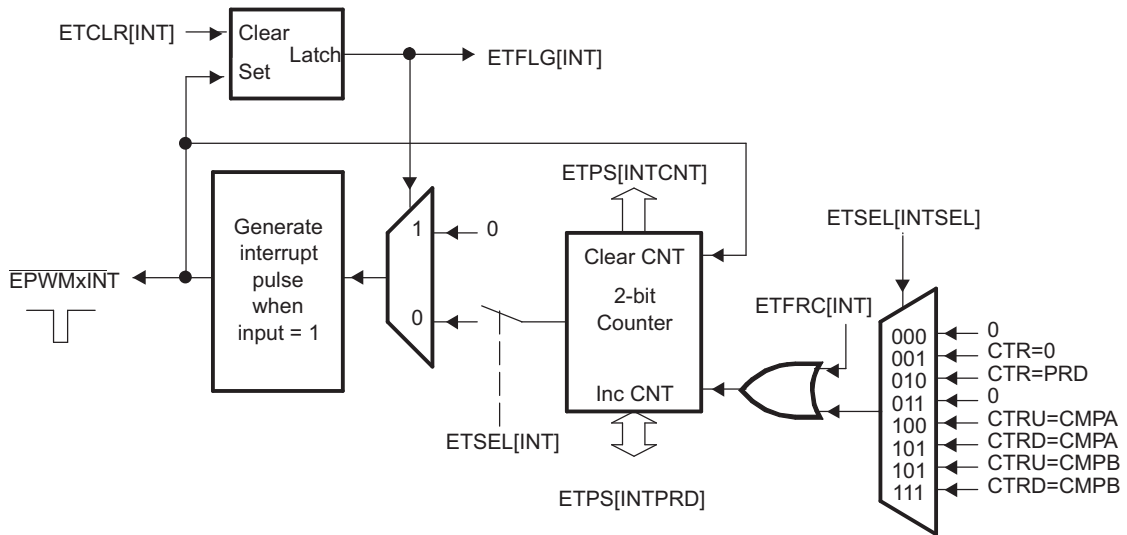
When ETPS[INTCNT] reaches ETPS[INTPRD], one of the following behaviors will occur:

- If interrupts are enabled, ETSEL[INTEN] = 1 and the interrupt flag is clear, ETFLG[INT] = 0, then an interrupt pulse is generated and the interrupt flag is set, ETFLG[INT] = 1, and the event counter is cleared ETPS[INTCNT] = 0. The counter will begin counting events again.
- If interrupts are disabled, ETSEL[INTEN] = 0, or the interrupt flag is set, ETFLG[INT] = 1, the counter stops counting events when it reaches the period value ETPS[INTCNT] = ETPS[INTPRD].
- If interrupts are enabled, but the interrupt flag is already set, then the counter will hold its output high until the ENTFLG[INT] flag is cleared. This allows for one interrupt to be pending while one is serviced.

Writing to the INTPRD bits will automatically clear the counter INTCNT = 0 and the counter output will be reset (so no interrupts are generated). Writing a 1 to the ETFRC[INT] bit will increment the event counter INTCNT. The counter will behave as described above when INTCNT = INTPRD. When INTPRD = 0, the counter is disabled and hence no events will be detected and the ETFRC[INT] bit is also ignored.

Note that the interrupts coming from the ePWM module are also used as DMA events. The interrupt registers should be used to enable and clear the current DMA event in order for the ePWM module to generate subsequent DMA events.

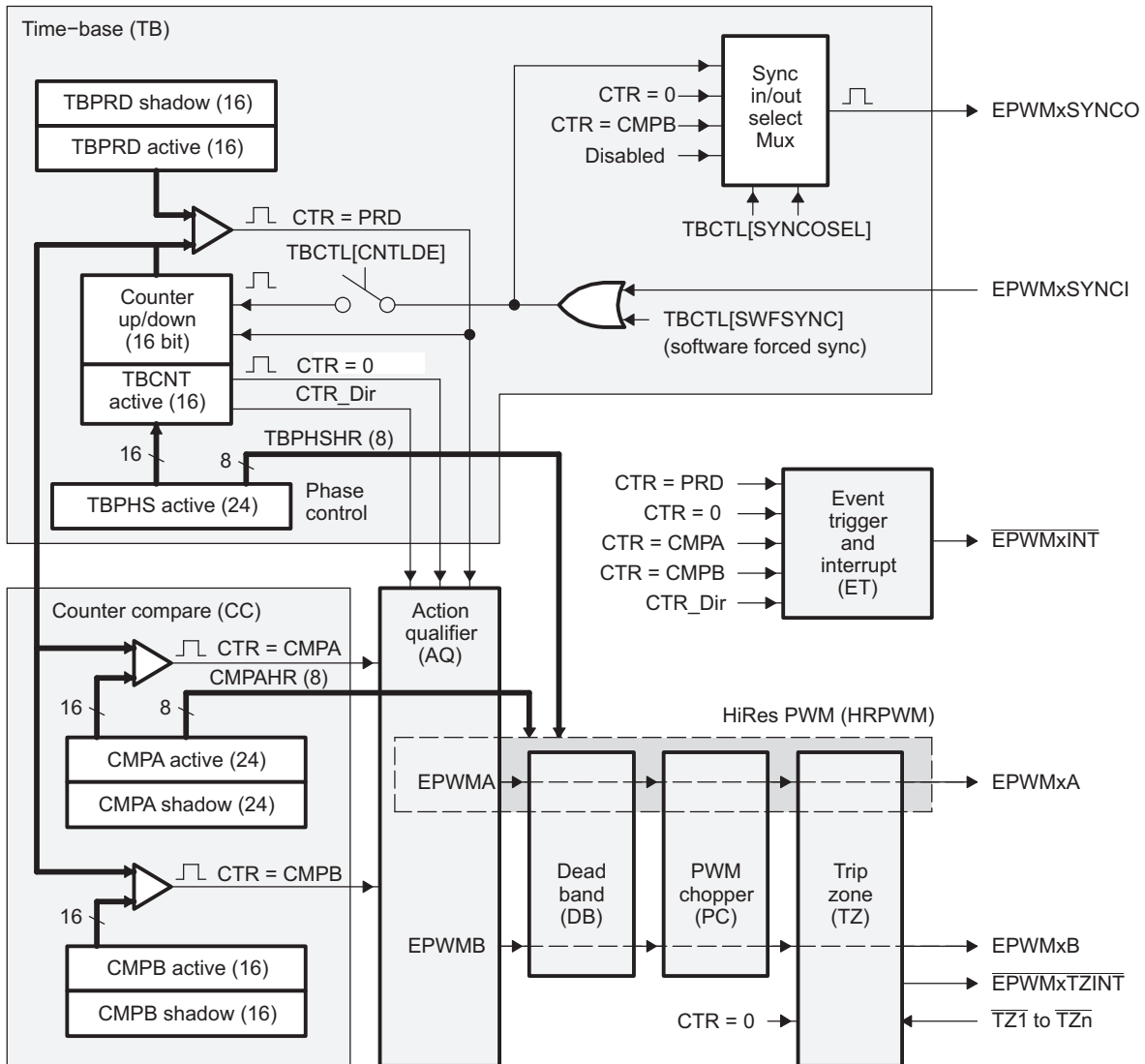
Figure 15-48. Event-Trigger Interrupt Generator



15.2.2.10 High-Resolution PWM (HRPWM) Submodule

Figure 15-49 shows the high-resolution PWM (HRPWM) submodule in the ePWM system. Some devices include the high-resolution PWM submodule, see Section 15.1.2 to determine which ePWM instances include this feature.

Figure 15-49. HRPWM System Interface



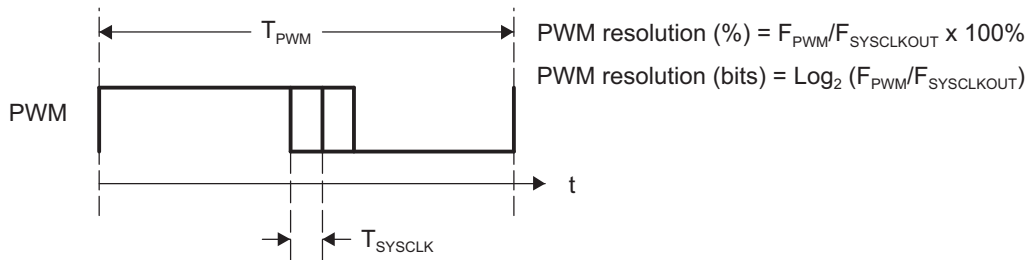
15.2.2.10.1 Purpose of the High-Resolution PWM Submodule

The enhanced high-resolution pulse-width modulator (eHRPWM) extends the time resolution capabilities of the conventionally derived digital pulse-width modulator (PWM). HRPWM is typically used when PWM resolution falls below ~9-10 bits. The key features of HRPWM are:

- Extended time resolution capability
- Used in both duty cycle and phase-shift control methods
- Finer time granularity control or edge positioning using extensions to the Compare A and Phase registers
- Implemented using the A signal path of PWM, that is, on the EPWMxA output. EPWMxB output has conventional PWM capabilities

The ePWM peripheral is used to perform a function that is mathematically equivalent to a digital-to-analog converter (DAC). As shown in Figure 15-50, the effective resolution for conventionally generated PWM is a function of PWM frequency (or period) and system clock frequency.

Figure 15-50. Resolution Calculations for Conventionally Generated PWM



If the required PWM operating frequency does not offer sufficient resolution in PWM mode, you may want to consider HRPWM. As an example of improved performance offered by HRPWM, Table 15-39 shows resolution in bits for various PWM frequencies. Table 15-39 values assume a MEP step size of 180 ps. See your device-specific data manual for typical and maximum performance specifications for the MEP.

Table 15-39. Resolution for PWM and HRPWM

PWM Frequency (kHz)	Regular Resolution (PWM)		High Resolution (HRPWM)	
	Bits	%	Bits	%
20	12.3	0.0	18.1	0.000
50	11.0	0.0	16.8	0.001
100	10.0	0.1	15.8	0.002
150	9.4	0.2	15.2	0.003
200	9.0	0.2	14.8	0.004
250	8.6	0.3	14.4	0.005
500	7.6	0.5	13.8	0.007
1000	6.6	1.0	12.4	0.018
1500	6.1	1.5	11.9	0.027
2000	5.6	2.0	11.4	0.036

Although each application may differ, typical low-frequency PWM operation (below 250 kHz) may not require HRPWM. HRPWM capability is most useful for high-frequency PWM requirements of power conversion topologies such as:

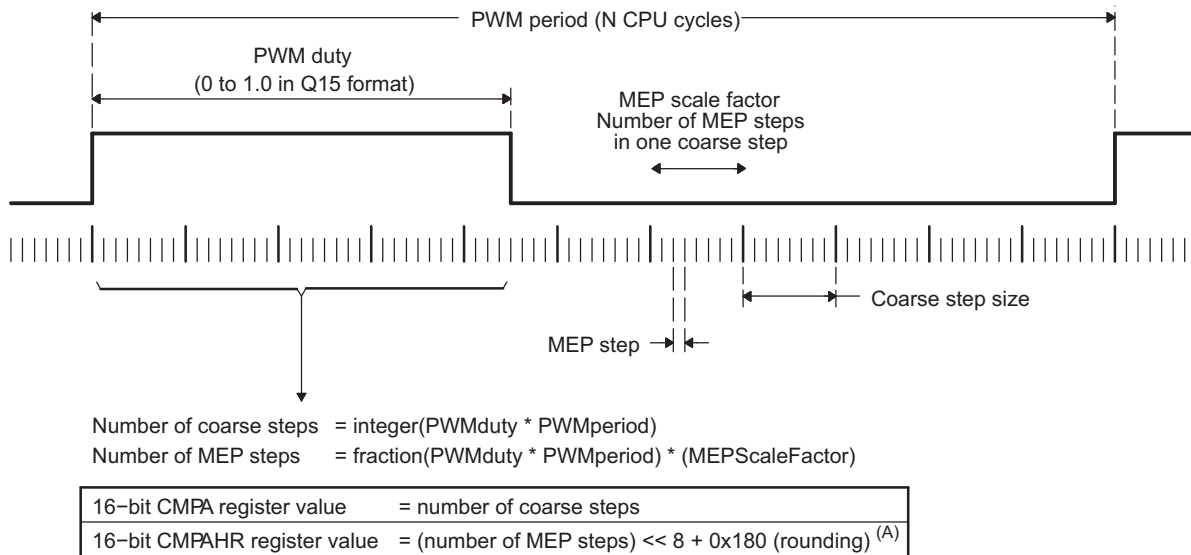
- Single-phase buck, boost, and flyback
- Multi-phase buck, boost, and flyback
- Phase-shifted full bridge
- Direct modulation of D-Class power amplifiers

15.2.2.10.2 Architecture of the High-Resolution PWM Submodule

The HRPWM is based on micro edge positioner (MEP) technology. MEP logic is capable of positioning an edge very finely by sub-dividing one coarse system clock of a conventional PWM generator. The time step accuracy is on the order of 150 ps. The HRPWM also has a self-check software diagnostics mode to check if the MEP logic is running optimally, under all operating conditions.

Figure 15-51 shows the relationship between one coarse system clock and edge position in terms of MEP steps, which are controlled via an 8-bit field in the Compare A extension register (CMPAHR).

Figure 15-51. Operating Logic Using MEP



A For MEP range and rounding adjustment.

To generate an HRPWM waveform, configure the TBM, CCM, and AQM registers as you would to generate a conventional PWM of a given frequency and polarity. The HRPWM works together with the TBM, CCM, and AQM registers to extend edge resolution, and should be configured accordingly. Although many programming combinations are possible, only a few are needed and practical.

15.2.2.10.3 Controlling and Monitoring the High-Resolution PWM Submodule

The MEP of the HRPWM is controlled by two extension registers, each 8-bits wide. These two HRPWM registers are concatenated with the 16-bit TBPHS and CMPA registers used to control PWM operation.

- TBPHSHR - Time-Base Phase High-Resolution Register
- CMPAHR - Counter-Compare A High-Resolution Register

Table 15-40 lists the registers used to control and monitor the high-resolution PWM submodule.

Table 15-40. HRPWM Submodule Registers

Acronym	Register Description	Address Offset	Shadowed
TBPHSHR	Extension Register for HRPWM Phase	4h	No
CMPAHR	Extension Register for HRPWM Duty	10h	Yes
HRCNFG	HRPWM Configuration Register	C0h	No

15.2.2.10.4 Configuring the High-Resolution PWM Submodule

Once the ePWM has been configured to provide conventional PWM of a given frequency and polarity, the HRPWM is configured by programming the HRCNFG register located at offset address C0h. This register provides configuration options for the following key operating modes:

- **Edge Mode:** The MEP can be programmed to provide precise position control on the rising edge (RE), falling edge (FE), or both edges (BE) at the same time. FE and RE are used for power topologies requiring duty cycle control, while BE is used for topologies requiring phase shifting, for example, phase shifted full bridge.
- **Control Mode:** The MEP is programmed to be controlled either from the CMPAHR register (duty cycle control) or the TBPMSHR register (phase control). RE or FE control mode should be used with CMPAHR register. BE control mode should be used with TBPMSHR register.
- **Shadow Mode:** This mode provides the same shadowing (double buffering) option as in regular PWM mode. This option is valid only when operating from the CMPAHR register and should be chosen to be the same as the regular load option for the CMPA register. If TBPMSHR is used, then this option has no effect.

15.2.2.10.5 Operational Highlights for the High-Resolution PWM Submodule

The MEP logic is capable of placing an edge in one of 255 (8 bits) discrete time steps, each of which has a time resolution on the order of 150 ps. The MEP works with the TBM and CCM registers to be certain that time steps are optimally applied and that edge placement accuracy is maintained over a wide range of PWM frequencies, system clock frequencies and other operating conditions. [Table 15-41](#) shows the typical range of operating frequencies supported by the HRPWM.

Table 15-41. Relationship Between MEP Steps, PWM Frequency and Resolution

System (MHz)	MEP Steps Per SYSCLKOUT ^{(1) (2) (3)}	PWM Minimum (Hz) ⁽⁴⁾	PWM Maximum (MHz)	Resolution at Maximum (Bits) ⁽⁵⁾
50.0	111	763	2.50	11.1
60.0	93	916	3.00	10.9
70.0	79	1068	3.50	10.6
80.0	69	1221	4.00	10.4
90.0	62	1373	4.50	10.3
100.0	56	1526	5.00	10.1

⁽¹⁾ System frequency = SYSCLKOUT, that is, CPU clock. TBCLK = SYSCLKOUT

⁽²⁾ Table data based on a MEP time resolution of 180 ps (this is an example value)

⁽³⁾ MEP steps applied = $T_{\text{SYSCLKOUT}}/180 \text{ ps}$ in this example.

⁽⁴⁾ PWM minimum frequency is based on a maximum period value, TBPRD = 65 535. PWM mode is asymmetrical up-count.

⁽⁵⁾ Resolution in bits is given for the maximum PWM frequency stated.

15.2.2.10.5.1 Edge Positioning

In a typical power control loop (switch modes, digital motor control (DMC), uninterruptible power supply (UPS)), a digital controller (PID, 2pole/2zero, lag/lead, etc.) issues a duty command, usually expressed in a per unit or percentage terms.

In the following example, assume that for a particular operating point, the demanded duty cycle is 0.405 or 40.5% on-time and the required converter PWM frequency is 1.25 MHz. In conventional PWM generation with a system clock of 100 MHz, the duty cycle choices are in the vicinity of 40.5%. In [Figure 15-52](#), a compare value of 32 counts (duty = 40%) is the closest to 40.5% that you can attain. This is equivalent to an edge position of 320 ns instead of the desired 324 ns. This data is shown in [Table 15-42](#).

By utilizing the MEP, you can achieve an edge position much closer to the desired point of 324 ns. [Table 15-42](#) shows that in addition to the CMPA value, 22 steps of the MEP (CMPAHR register) will position the edge at 323.96 ns, resulting in almost zero error. In this example, it is assumed that the MEP has a step resolution of 180 ns.

Figure 15-52. Required PWM Waveform for a Requested Duty = 40.5%

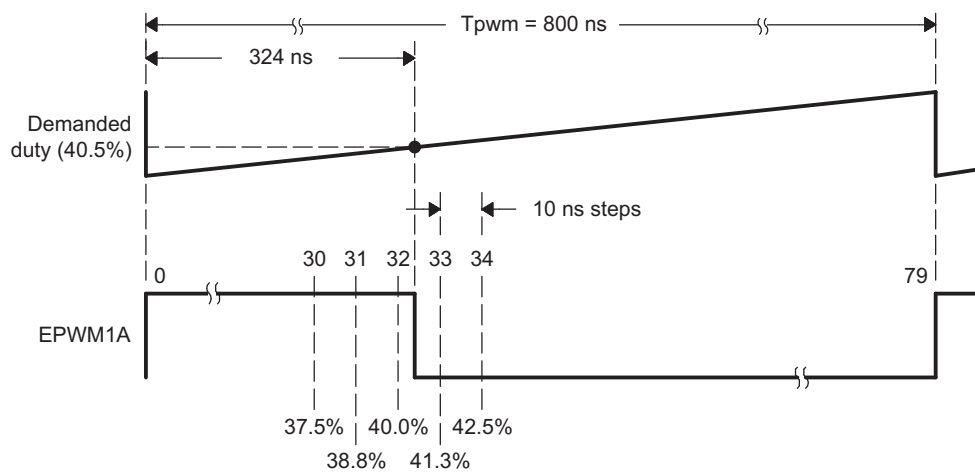


Table 15-42. CMPA vs Duty (left), and [CMPA:CMPAHR] vs Duty (right)

CMPA (count) ^{(1) (2) (3)}	DUTY (%)	High Time (ns)	CMPA (count)	CMPAHR (count)	Duty (%)	High Time (ns)
28	35.0	280	32	18	40.405	323.24
29	36.3	290	32	19	40.428	323.42
30	37.5	300	32	20	40.450	323.60
31	38.8	310	32	21	40.473	323.78
32	40.0	320	32	22	40.495	323.96
33	41.3	330	32	23	40.518	324.14
34	42.5	340	32	24	40.540	324.32
			32	25	40.563	324.50
Required			32	26	40.585	324.68
32.40	40.5	324	32	27	40.608	324.86

⁽¹⁾ System clock, SYSCLKOUT and TBCLK = 100 MHz, 10 ns

⁽²⁾ For a PWM Period register value of 80 counts, PWM Period = 80 × 10 ns = 800 ns, PWM frequency = 1/800 ns = 1.25 MHz

⁽³⁾ Assumed MEP step size for the above example = 180 ps

15.2.2.10.5.2 Scaling Considerations

The mechanics of how to position an edge precisely in time has been demonstrated using the resources of the standard (CMPA) and MEP (CMPAHR) registers. In a practical application, however, it is necessary to seamlessly provide the CPU a mapping function from a per-unit (fractional) duty cycle to a final integer (non-fractional) representation that is written to the [CMPA:CMPAHR] register combination.

To do this, first examine the scaling or mapping steps involved. It is common in control software to express duty cycle in a per-unit or percentage basis. This has the advantage of performing all needed math calculations without concern for the final absolute duty cycle, expressed in clock counts or high time in ns. Furthermore, it makes the code more transportable across multiple converter types running different PWM frequencies.

To implement the mapping scheme, a two-step scaling procedure is required.

Assumptions for this example:

System clock, SYSCLKOUT	= 10 ns (100 MHz)
PWM frequency	= 1.25 MHz (1/800 ns)
Required PWM duty cycle, PWMDuty	= 0.405 (40.5%)
PWM period in terms of coarse steps, PWMperiod (800 ns/10 ns)	= 80
Number of MEP steps per coarse step at 180 ps (10 ns/180 ps), MEP_SF	= 55
Value to keep CMPAHR within the range of 1-255 and fractional rounding constant (default value)	= 180h

Step 1: Percentage Integer Duty value conversion for CMPA register

CMPA register value	= $\text{int}(\text{PWMDuty} \times \text{PWMperiod})$; int means integer part
	= $\text{int}(0.405 \times 80)$
	= $\text{int}(32.4)$
CMPA register value	= 32 (20h)

Step 2: Fractional value conversion for CMPAHR register

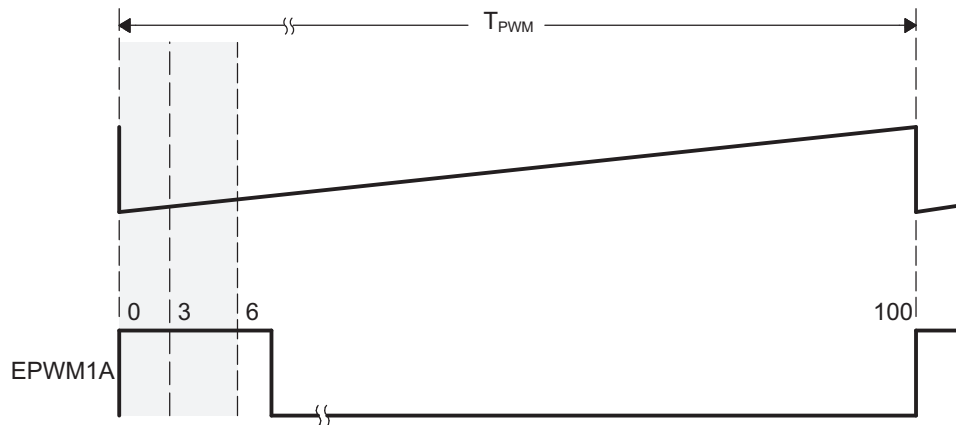
CMPAHR register value	= $(\text{frac}(\text{PWMDuty} \times \text{PWMperiod}) \times \text{MEP_SF}) \ll 8) + 180\text{h}$; frac means fractional part
	= $(\text{frac}(32.4) \times 55 \ll 8) + 180\text{h}$; Shift is to move the value as CMPAHR high byte
	= $((0.4 \times 55) \ll 8) + 180\text{h}$
	= $(22 \ll 8) + 180\text{h}$
	= $22 \times 256 + 180\text{h}$; Shifting left by 8 is the same multiplying by 256.
	= $5632 + 180\text{h}$
	= $1600\text{h} + 180\text{h}$
CMPAHR value	= 1780h; CMPAHR value = 1700h, lower 8 bits will be ignored by hardware.

15.2.2.10.5.3 Duty Cycle Range Limitation

In high resolution mode, the MEP is not active for 100% of the PWM period. It becomes operational 3 SYCLK cycles after the period starts.

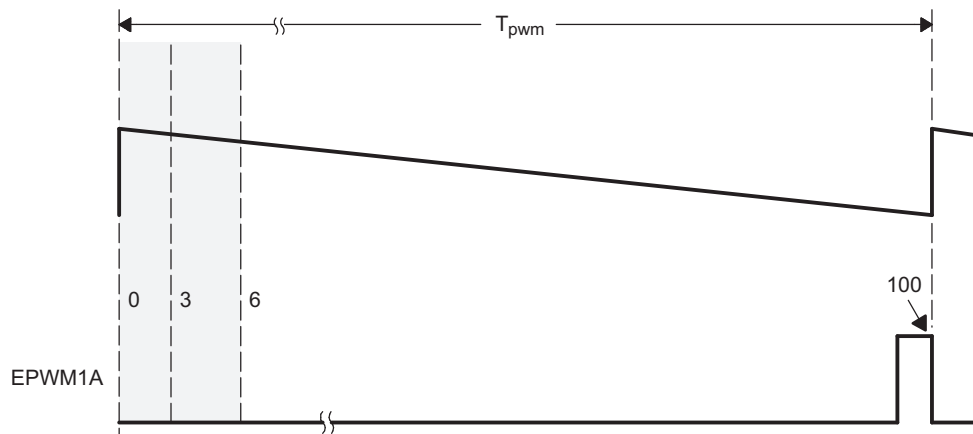
Duty cycle range limitations are illustrated in [Figure 15-53](#). This limitation imposes a lower duty cycle limit on the MEP. For example, precision edge control is not available all the way down to 0% duty cycle. Although for the first 3 or 6 cycles, the HRPWM capabilities are not available, regular PWM duty control is still fully operational down to 0% duty. In most applications this should not be an issue as the controller regulation point is usually not designed to be close to 0% duty cycle.

Figure 15-53. Low % Duty Cycle Range Limitation Example When PWM Frequency = 1 MHz



If the application demands HRPWM operation in the low percent duty cycle region, then the HRPWM can be configured to operate in count-down mode with the rising edge position (REP) controlled by the MEP. This is illustrated in [Figure 15-54](#). In this case low percent duty limitation is no longer an issue.

Figure 15-54. High % Duty Cycle Range Limitation Example when PWM Frequency = 1 MHz



15.2.2.10.6 Scale Factor Optimizing Software (SFO)

The micro edge positioner (MEP) logic is capable of placing an edge in one of 255 discrete time steps. As previously mentioned, the size of these steps is on the order of 150 ps. The MEP step size varies based on worst-case process parameters, operating temperature, and voltage. MEP step size increases with decreasing voltage and increasing temperature and decreases with increasing voltage and decreasing temperature. Applications that use the HRPWM feature should use the TI-supplied MEP scale factor optimizer (SFO) software functions. SFO functions help to dynamically determine the number of MEP steps per SYSCLKOUT period while the HRPWM is in operation.

To utilize the MEP capabilities effectively during the Q15 duty to [CMPA:CMPAHR] mapping function (see [Section 15.2.2.10.5.2](#)), the correct value for the MEP scaling factor (MEP_SF) needs to be known by the software. To accomplish this, each HRPWM module has built in self-check and diagnostics capabilities that can be used to determine the optimum MEP_SF value for any operating condition. TI provides a C-callable library containing two SFO functions that utilize this hardware and determine the optimum MEP_SF. As such, MEP Control and Diagnostics registers are reserved for TI use.

Currently, there are two released versions of the SFO library - SFO_TI_Build.lib and SFO_TI_Build_V5.lib. Versions 2, 3, and 4 were TI-Internal only. A high-level comparison table between the two library versions can be found in [Section 15.2.2.10.6.1](#). A detailed description of the SFO_TI_Build.lib software functions follows in [Section 15.2.2.10.6.2](#). Information on the SFO_TI_Build_V5.lib software functions, which support up to 16 HRPWM channels, can be found in [Section 15.2.2.10.6.4](#).

NOTE: For floating-point devices, when compiling application code for floating point (fpu32 mode), libraries utilized by the application code must also be compiled for floating point. The SFO_TI_Build_fpu.lib and SFO_TI_Build_V5_fpu.lib are available as the floating-point compiled equivalents to the fixed-point SFO_TI_Build.lib and SFO_TI_Build_V5.lib libraries. The SFO functions in the fpu-version libraries are C-code-compatible to their fixed-point equivalents.

15.2.2.10.6.1 SFO Library Version Comparison

[Table 15-43](#) includes a high-level comparison between SFO_TI_Build.lib and SFO_TI_Build_V5.lib. A detailed description of SFO_TI_Build_V5.lib follows the table, and more information on SFO_TI_Build.lib can be found in [Section 15.2.2.10.6](#).

Table 15-43. SFO Library Version Comparison

	SYSCLK Frequency	ePWM Frequency	SFO_TI_Build.lib	SFO_TI_Build_V5.lib	Unit
Maximum HRPWM channels supported	-	-	Up to 4	Up to 16	channels
Total static variable memory size	-	-	220	79 (1 channel) to 192 (16 channel)	words
MepEn runs on multiple channels concurrently?	-	-	Yes	No	-
Error-checking?	-	-	No	Yes	-
Typical time required for MepEn to update	-	400 kHz	3.26	1.5	seconds
MEP_ScaleFactor on 1 channel if called repetitively without interrupts	-	1 MHz	1.308	0.6	seconds
	-	2 MHz	0.66	0.3	seconds
	-	3.33 MHz	0.396	0.18	seconds
	-	200 MHz	0.066	0.03	seconds
Typical time required for MepDis to update	100 MHz	-	0.83	1.86	milliseconds
MEP_ScaleFactor on 1 channel if called repetitively without interrupts	60 MHz	-	1.38	1.38	milliseconds
	50 MHz	-	1.66	1.66	milliseconds

15.2.2.10.6.2 SFO_TI_Build Library Routines

Table 15-44 provides functional descriptions of the two SFO library routines in SFO_TI_Build.lib.

Table 15-44. SFO Library Routines

Function	Description
SFO_MepDis(n)	<p>Scale Factor Optimizer with MEP Disabled</p> <p>This routine runs faster, as the calibration logic works when HRPWM capabilities are disabled; therefore, HRPWM capabilities cannot be run concurrently when the ePWMn is being used.</p> <p>The function returns a value in the variable array:</p> $\text{MEP_ScaleFactor}[n]^{(1)} = \text{Number of MEP steps/SYSCLKOUT}$ <p>If TBCLK is not equal to SYSCLKOUT, then the returned value must be adjusted to reflect the correct TBCLK:</p> $\text{MEP steps per TBCLK} = \text{MEP_ScaleFactor}[n] \times (\text{SYSCLKOUT}/\text{TBCLK})^{(1)}$ <p>Constraints when using this function:</p> <p>MEP diagnostics logic uses SYSCLKOUT not TBCLK and hence SYSCLKOUT restriction is an important constraint. SFO_MepDis(n) function does not require a starting Scale Factor value.</p> <p>When to use:</p> <p>If one of the ePWM modules is not used in HRPWM mode, then it can be dedicated to run the SFO diagnostics for the modules that are running HRPWM mode. Here the single MEP_SF value obtained can be applied to other ePWM modules. This assumes that all HRPWM module's MEP steps are similar but may not be identical.</p> <p>The ePWM module that is not active in HRPWM mode is still fully operational in conventional PWM mode and can be used to drive PWM pins. The SFO function only makes use of the MEP diagnostics logic.</p> <p>The other ePWM modules operating in HRPWM mode incur only a 3-cycle minimum duty limitation.</p>
SFO_MepEn(n)	<p>Scale Factor Optimizer with MEP Enabled</p> <p>This routine runs slower as the calibration logic is used concurrently while HRPWM capabilities are being used by the ePWM module.</p> <p>The function returns a value in the variable array:</p> $\text{MEP_ScaleFactor}[n]^{(1)} = \text{Number of MEP steps/SYSCLKOUT}$ $= \text{Number of MEP steps/TBCLK}$ <p>Constraints when using this function:</p> <p>MEP diagnostics logic uses SYSCLKOUT not TBCLK and hence SYSCLKOUT restriction is an important constraint. SFO_MepEn(n) function does require a starting Scale Factor value. MEP_ScaleFactor[0] needs to be initialized to a typical MEP step size value.</p> <hr/> <p>NOTE: SFO_MepEn(n) only supports the following HRPWM configuration:</p> <ul style="list-style-type: none"> • HRCNFG[HRLOAD] = 0 (load on CTR = 0) • HRCNFG[EDGMODE] = 10 (falling edge MEP control) <hr/> <p>When to use:</p> <p>If the application requires all ePWM modules to have HRPWM capability (MEP is operational), then the SFO_MepEn(n) function should run for each of the active ePWM modules with HRPWM capability.</p> <ul style="list-style-type: none"> • In the above case, a 6-cycle MEP inactivity zone exists at the start of the PWM period. See Section 15.2.2.10.5.3 on duty cycle range limitation. • If all ePWM modules are using the same TBCLK prescaler, then it is also possible to run the SFO_MepEn(n) function for only one ePWM module and to use the SFO return value for the other modules. In this case only one ePWM module incurs the 6-cycle limitation, and remaining modules incur only a 3-cycle minimum duty limitation. See "Duty cycle limitation" section. This assumes that all HRPWM module's MEP steps are similar but may not be identical.

⁽¹⁾ n is the ePWM module number on which the SFO function operates.

Both routines can be run as background tasks in a slow loop requiring negligible CPU cycles. In most applications only one of these routines will be needed. However, if the application has free HRPWM resources then both the routines could be used. The repetition rate at which an SFO function needs to be executed depends on the applications operating environment. As with all digital CMOS devices temperature and supply voltage variations have an effect on MEP operation. However, in most applications these parameters vary slowly and therefore it is often sufficient to execute the SFO function once every 5 to 10 seconds or so. If more rapid variations are expected, then execution may have to be performed more frequently to match the application. Note, there is no high limit restriction on the SFO function repetition rate, hence it can execute as quickly as the background loop is capable.

While using HRPWM feature with no SFO diagnostics, HRPWM logic will not be active for the first 3 TBCLK cycles of the PWM period. While running the application in this configuration, if CMPA register value is less than 3 cycles, then its CMPAHR register must be cleared to zero. This would avoid any unexpected transitions on PWM signal.

However, if SFO diagnostic function SFO_MepEn is used in the background, then HRPWM logic will not be active for the first 6 TBCLK cycles of PWM period. While using SFO_MepEn function if CMPA register value is less than 6 cycles, then its CMPAHR register must be cleared to zero. This would avoid any unexpected transitions on PWM signal. Also note that the SFO_MepDis function cannot be used concurrently with PWM signals with HRPWM enabled.

15.2.2.10.6.3 SFO_TI_Build Software Usage

Software library functions SFO_MepEn(int n) and SFO_MepDis(int n) calculate the MEP scale factor for up to four ePWM modules. The scale factor is an integer value in the range 1–255, and represents the number of micro step edge positions available for a system clock period. The scale factor value is returned in a global array of integer values called MEP_ScaleFactor[x], where x is the maximum number of HRPWM channels for a device plus one. For example, see [Table 15-45](#).

Table 15-45. Factor Values

Software function calls	Functional description	Updated Variable MEP_ScaleFactor[5] ⁽¹⁾
SFO_MepDis(n)		
SFO_MepDis(1);	Returns the scale factor value to array index 1	MEP_ScaleFactor[1]
SFO_MepDis(2);	Returns the scale factor value to array index 2	MEP_ScaleFactor[2]
SFO_MepDis(3);	Returns the scale factor value to array index 3	MEP_ScaleFactor[3]
SFO_MepDis(4);	Returns the scale factor value to array index 4	MEP_ScaleFactor[4]
SFO_MepEn(n)		
SFO_MepEn(1);	Returns the scale factor value to array index 1	MEP_ScaleFactor[1]
SFO_MepEn(2);	Returns the scale factor value to array index 2	MEP_ScaleFactor[2]
SFO_MepEn(3);	Returns the scale factor value to array index 3	MEP_ScaleFactor[3]
SFO_MepEn(4);	Returns the scale factor value to array index 4	MEP_ScaleFactor[4]

⁽¹⁾ MEP_ScaleFactor[0] variable is a starting value and used by the SFO software functions internally

To use the HRPWM feature of the ePWMs it is recommended that the SFO functions be used as follows. For different devices that may have fewer HRPWM channels, modifications will be required in Step 1 and Step 2.

NOTE: The following example assumes there are four ePWM instances that contain the HRPWM submodule in the device. See [Section 15.1.2](#) to determine the number of ePWM instances that contain the HRPWM submodule.

Step 1. Add Include Files

The SFO.h file needs to be included as follows. This include file is mandatory while using the SFO library function. These include files are optional if customized header files are used in the end applications.

Example 15-3. A Sample of How to Add Include Files

```
#include "SFO.h"           // SFO lib functions (needed for HRPWM)
```

Step 2. Element Declaration

Declare a 5-element array of integer variables as follows:

Example 15-4. Declaring an Element

```
int    MEP_ScaleFactor[5] = {0,0,0,0,0}; // Scale factor values for ePWM1-4
int    MEP_SF1, MEP_SF2, MEP_SF3, MEP_SF4
volatile struct EPWM_REGS *ePWM[] = {&EPwm1Regs, &EPwm1Regs, &EPwm2Regs, &EPwm3Regs, &EPwm4Regs};
```

Step 3. MEP_ScaleFactor Initialization

After power up, the SFO_MepEn(n) function needs a starting Scale Factor value. This value can be conveniently determined by using one of the ePWM modules to run the SFO_MepDis(n) function prior to configuring its PWM outputs for the application. SFO_MepDis(n) function does not require a starting Scale Factor value.

As part of the one-time initialization code, include the following:

Example 15-5. Initializing With a Scale Factor Value

```
//    MEP_ScaleFactor variables initialized using function SFO_MepDis
while (MEP_ScaleFactor[1] == 0) SFO_MepDis(1); //SFO for HRPWM1
while (MEP_ScaleFactor[2] == 0) SFO_MepDis(2); //SFO for HRPWM2
while (MEP_ScaleFactor[3] == 0) SFO_MepDis(3); //SFO for HRPWM3
while (MEP_ScaleFactor[4] == 0) SFO_MepDis(4); //SFO for HRPWM4

// Initialize a common seed variable MEP_ScaleFactor[0] // required for all SFO functions
MEP_ScaleFactor[0] = MEP_ScaleFactor[1]; //Common variable for SFOMepEN(n) function
```

Step 4. Application Code

While the application is running, fluctuations in both device temperature and supply voltage may be expected. To be sure that optimal Scale Factors are used for each ePWM module, the SFO function should be re-run periodically as part of a slower back-ground loop. Some examples of this are shown here.

Example 15-6. SFO Function Calls

```
main()
{
    // User code
    // Case1: ePWM1,2,3,4 are running in HRPWM mode
    SFO_MepEn(1);           // Each of these of function enables
    SFO_MepEn(2);           // the respective MEP diagnostic logic
    SFO_MepEn(3);           // and returns MEP Scale factor value
    SFO_MepEn(4);

    MEP_SF1 = MEP_ScaleFactor[1]; // used for ePWM1
    MEP_SF2 = MEP_ScaleFactor[2]; // used for ePWM2
    MEP_SF3 = MEP_ScaleFactor[3]; // used for ePWM3
    MEP_SF4 = MEP_ScaleFactor[4]; // used for ePWM4

    // Case2:ePWM1,2,3 only are running in HRPWM mode.
    // One of the ePWM channel(as an example ePWM4) is used as for
    // Scale factor calibration

    // Here minimum duty cycle limitation is 3 clock cycles.
    //
    // HRPWM 4 MEP diagnostics circuit is used to estimate the MEP steps
    // with the assumption that all HRPWM channels behave similarly
    // though may not be identical.

    SFO_MepDis(4);         // MEP steps using ePWM4
    MEP_SF1 = MEP_ScaleFactor[4]; // used for ePWM1
    MEP_SF2 = MEP_SF1      // used for ePWM2
    MEP_SF3 = MEP_SF1      // used for ePWM3
    MEP_SF4 = MEP_SF1      // used for ePWM4
}
```

15.2.2.10.6.4 SFO_TI_Build_V5 Library Routines

In SFO_TI_Build_V5.lib, the diagnostic software has been optimized to use less memory, to minimize the calibration time, and to support up to 16 HRPWM channels. [Table 15-46](#) provides functional description of the two SFO library routines in SFO_TI_Build_V5.lib.

Table 15-46. SFO V5 Library Routines

Function	Description
int SFO_MepDis_V5 (n)	<p>Scale Factor Optimizer V5 with MEP Disabled</p> <p>This routine is very similar to the SFO_MepDis() routine in the original SFO library, but with one change. It now returns a 1 when MEP disabled calibration is complete, or a 0 while calibration is still running.</p> <p>This function runs faster than the SFO_MepEn_V5() routine and cannot be used on an ePWM channel while HRPWM capabilities are enabled for that channel. If there is a spare ePWM channel available in the system, SFO_MepDis_V5() can be run for that channel, and the resulting MEP_ScaleFactor[n] value can be copied into the MEP_ScaleFactor[n] for all other channels.</p> <p>Because the MEP step behavior in a particular piece of silicon is similar on all HRPWM channels with regard to temperature and voltage, using one dedicated HRPWM channel for calibration by calling the SFO_MepDis_V5 function will reduce software overhead.</p> <p>The function returns a value in the variable array:</p> $\text{MEP_ScaleFactor}[n] = \text{Number of MEP steps/SYSCLKOUT}$ <p>If TBCLK is not equal to SYSCLKOUT, then the returned value must be adjusted to reflect the correct TBCLK:</p> $\text{MEP steps per TBCLK} = \text{MEP_ScaleFactor}[n] \times (\text{SYSCLKOUT}/\text{TBCLK})$ <p>Constraints when using this function:</p> <ul style="list-style-type: none"> • MEP diagnostics logic uses SYSCLKOUT and not TBCLK. Hence, the SYSCLKOUT restriction is an important constraint. • If TBCLK does not equal SYSCLKOUT, the TBCLK frequency must be great enough so that MEP steps per TBCLK do not exceed 255. This is due to the restriction that there can be no more than 255 MEP steps in a coarse step. • This function cannot be run on an ePWM channel with HRPWM capabilities enabled. <p>Usage:</p> <ul style="list-style-type: none"> • If one of the ePWM modules is running in normal ePWM mode, then it can be used to run the SFO diagnostics function. Here, the single MEP_ScaleFactor value obtained for that channel can be copied and used as the MEP_ScaleFactor for the other ePWM modules which are running HRPWM mode. This assumes that all HRPWM modules' MEP steps are similar but may not be identical. • This routine returns a 1 when calibration is finished on the specified channel or a 0 if calibration is still running. • The ePWM module that is not active in HRPWM mode is still fully operational in conventional PWM mode and can be used to drive PWM pins. The SFO function only makes use of the MEP diagnostics logic in the HRPWM circuitry. • SFO_MepDis_V5(n) function does not require a starting Scale Factor value. • The other ePWM modules operating in HRPWM mode incur only a 3-cycle minimum duty cycle limitation.

Table 15-46. SFO V5 Library Routines (continued)

Function	Description
int SFO_MepEn_V5 (n)	<p>Scale Factor Optimizer V5 with MEP Enabled</p> <p>This function runs slower than the SFO_MepDis_V5() routine and runs SFO diagnostics on an ePWM channel with HRPWM capabilities enabled for that channel.</p> <p>The function returns a value in the variable array:</p> <p style="margin-left: 40px;">MEP_ScaleFactor[n] = Number of MEP steps/SYSCLKOUT = Number of MEP steps/TBCLK</p> <p>Constraints when using this function:</p> <ul style="list-style-type: none"> This routine must be run on one channel at a time and cannot be run on multiple channels concurrently. When it has finished updating the MEP_ScaleFactor for a channel, it will return a 1. If it is still calibrating, it will return a 0. A background loop should exist in the user code which calls SFO_MepEn_V5(n) repetitively until it returns a 1. Then the function can be called for the next channel.⁽¹⁾ <hr/> <p>NOTE: Unlike the original SFO_MepEn(n) routine, this routine cannot run on multiple channels concurrently.</p> <p>Do not call SFO_MepEn_V5(n) for another channel until the function returns a 1 for the current channel. Otherwise, the MEP_ScaleFactor for both channels will become corrupted.</p> <p>SFO_MepEn_V5(n) in <i>SFO_TI_Build_V5.lib</i> supports only the following HRPWM configuration:</p> <ul style="list-style-type: none"> HRCNFG[HRLOAD] = 0 (load on CTR = 0) HRCNFG[EDGMODE] = 10 (falling edge MEP control) <p>An upgraded version of SFO_MepEn_V5(n) in <i>SFO_TI_Build_V5B.lib</i> supports all available HRPWM configurations. When using this version, the HRCNFG register must be initialized with the appropriate configuration after calling SFO_MepDis_V5(n) to seed the MEP_ScaleFactor[n] and prior to calling SFO_MepEn_V5(n).</p> <hr/> <ul style="list-style-type: none"> MEP diagnostics logic uses SYSCLKOUT and not TBCLK. Hence, the SYSCLKOUT restriction is an important constraint. <p>Usage:</p> <ul style="list-style-type: none"> After calling SFO_MepDis(n) to seed MEP_ScaleFactor[n], and prior to using the SFO_MepEn_V5(n) function in <i>SFO_TI_Build_V5B.lib</i>, the HRCNFG register must be initialized with the desired HRPWM configuration. Otherwise, calibration will not be initiated, and calls to SFO_MepEn_V5(n) will continuously return zero. The SFO_MepEn_V5(n) function requires a starting scale factor value, MEP_ScaleFactor[0]. MEP_ScaleFactor[0] needs to be initialized to a typical MEP step size value. To do this, SFO_MepDis_V5(n) can be run on an ePWM channel while the HRPWM is disabled, and the resulting MEP_ScaleFactor[n] value can be copied into MEP_ScaleFactor[0]. If there are drastic environmental changes to your system (i.e. temperature/voltage), it is generally a good idea to re-seed MEP_ScaleFactor[0] with a new typical MEP step size value for the changed conditions. Because SFO_MepEn_V5(n) can be run on only one channel at a time, it is only recommended for systems where there are no spare HRPWM channels available, so SFO calibration must be performed on all channels with HRPWM capabilities enabled. In this case, a 6-cycle MEP inactivity zone exists at the start of each PWM period on all HRPWM channels. See Section 15.2.2.10.5.3 on duty cycle range limitation. The function returns: <ul style="list-style-type: none"> A 1 when it has finished SFO calibration for the current channel A 0 when SFO diagnostics are still running for the channel A 2 as an error indicator after calibration has completed if the resulting MEP_ScaleFactor for <p>⁽¹⁾ If SFO calibration must be run on multiple channels at a time while HRPWM capabilities are enabled, the previous version of the SFO library, <i>SFO_TI_Build.lib</i>, which uses more memory resources, can be used instead, and SFO_MepEn(n) can run concurrently for up to 4 ePWM channels with HRPWM enabled.</p>

Table 15-46. SFO V5 Library Routines (continued)

Function	Description
	<p>the channel differs from the original MEP_ScaleFactor[0] seed value by more than +/- 15. The function must be called repetitively before it will return a 1. This takes a longer time to complete than the SFO_MepDis_V5(n) calibration.</p> <p>If it returns a 2, the MEP_ScaleFactor for the channel has finished updating and is outside the typical drift range of MEP_ScaleFactor[0] +/-15 even with large temperature and voltage variations. If the reason for the large difference between the seed and the channel scale factor is known and acceptable, the user may choose to ignore the return of 2, and treat it as a return value of 1, indicating that calibration is complete.</p> <p>Otherwise, if the large difference is unexpected, there are steps to take to remedy the error:</p> <ol style="list-style-type: none"> 1. Check your code to ensure SFO_MepEn_V5(n) is not being called on more than one channel at a time. 2. If the above is not effective, run SFO_MepDis_V5(n) again and re-seed Mep_ScaleFactor[0]. 3. If neither of the above 2 steps work, there may be a system problem. The application firmware should perform a shutdown or an appropriate recovery procedure. <ul style="list-style-type: none"> • If all ePWM modules are using the same TBCLK prescaler, then it is possible to run the SFO_MepEn_V5(n) function for only one ePWM module and to use the MEP_ScaleFactor value for that module for the other modules also. In this case only one ePWM module incurs the 6-cycle duty limitation, and the remaining modules incur only a 3-cycle minimum duty limitation. This assumes that all HRPWM modules' MEP steps are similar but may not be identical.

15.2.2.10.6.5 SFO_TI_Build_V5 Software Usage

Software library functions `int SFO_MepEn_V5(int n)` and `int SFO_MepDis_V5(int n)` calculate the MEP scale factor for ePWMn modules, where n= the ePWM channel number. The scale factor value, which represents the number of micro-steps available in a system clock period, is returned in a global array of integer values called `MEP_ScaleFactor[x]`, where x is the maximum number of HRPWM channels for a device plus one. For example, if the maximum number of HRPWM channels for a device is 16, the scale factor array would be `MEP_ScaleFactor[17]`. Both `SFO_MepEn_V5` and `SFO_MepDis_V5` themselves also return a 1 when calibration has completed, indicating the `MEP_ScaleFactor` has been successfully updated for the channel, and a 0 when calibration is still on-going. A return of 2 represents an out-of-range error.

Table 15-47. Software Functions

Software functional calls	Functional Description
int SFO_MepDis_V5(int n)⁽¹⁾	
<code>status = SFO_MepDis_V5(1)</code>	The scale factor in <code>MEP_ScaleFactor[1]</code> is updated when status = 1
<code>status = SFO_MepDis_V5(2)</code>	The scale factor in <code>MEP_ScaleFactor[2]</code> is updated when status = 1
...	...
<code>status = SFO_MepDis_V5(16)</code>	The scale factor in <code>MEP_ScaleFactor[16]</code> is updated when status = 1 or 2
int SFO_MepEn_V5(int n)⁽¹⁾	
<code>status = SFO_MepEn_V5(1);</code>	The scale factor in <code>MEP_ScaleFactor[1]</code> is updated when status = 1 or 2
<code>status = SFO_MepEn_V5(2);</code>	The scale factor in <code>MEP_ScaleFactor[2]</code> is updated when status = 1 or 2
...	...
<code>status = SFO_MepEn_V5(16);</code>	The scale factor in <code>MEP_ScaleFactor[16]</code> is updated when status = 1 or 2

⁽¹⁾ `MEP_ScaleFactor[0]` is a starting seed value used by the SFO software functions internally.

To use the HRPWM feature of the ePWMs, it is recommended that the SFO functions in `TI_Build_V5.lib` be used as described here. For different devices that may have fewer HRPWM channels, modifications will be required in Step 1 and Step 2.

NOTE: The following example assumes there are four ePWM instances that contain the HRPWM submodule in the device. See [Section 15.1.2](#) to determine the number of ePWM instances that contain the HRPWM submodule.

Step 1. Add Include Files

The `SFO_V5.h` file needs to be included as follows. This include file is mandatory when using the SFO V5 library functions.

Example 15-7. A Sample of How to Add Include Files

```
#include SFO_V5.h // SFO V5 lib functions (needed for HRPWM)
```

Step 2. Define Number of HRPWM Channels Used

In the `SFO_V5.h` file, the maximum number of HRPWM's used for a particular device must be defined. `PWM_CH` must equal the number of HRPWM channels plus 1.

To save static variable memory, fewer than the maximum number of HRPWM channels may be defined with some caution. To do this, `PWM_CH` can be set to the largest ePWM channel number plus 1. For instance, if only ePWM1A and ePWM2A channels are required as HRPWM channels, `PWM_CH` can be set to 3. However, if only ePWM15A and ePWM16A channels are required as HRPWM channels, `PWM_CH` must still be set to 17.

Example 15-8. Defining Number of HRPWM Channels Used (Plus 1)

```
// SFO_V5.H
// NOTE: THIS IS A VERY IMPORTANT STEP. PWM_CH MUST BE DEFINED FIRST BEFORE
// BUILDING CODE.

#define PWM_CH 17 // Maximum of 16 HRPWM channels (16+1=17)
```

Step 3. Element Declaration

Declare an array of integer variables with a length equal to PWM_CH, and an array of pointers to EPWM register structures. The array of pointers will include pointers for up to 16 EPWM register structures plus one dummy pointer in location EPWM[0] for a device with 16 EPWM channels. Likewise, it will include pointers for up to 4 EPWM register structures plus 1 for a device with 4 EPWM registers and up to 3 EPWM register structures plus 1 for a device with 3 EPWM registers.

Example 15-9. Declaring Elements Required by SFO_TI_Build_V5.lib

```
int MEP_ScaleFactor[PWM_CH] = {0,0,0,0,0, // Scale factor values for ePWM 1-16
                               0,0,0,0, // and MEP_ScaleFactor[0]
                               0,0,0,0, // For fewer HRPWM channels, there
                               0,0,0,0}; // will be fewer zeros initialized.

// Declare a volatile array of pointers to EPWM Register structures.
// Only point to registers that exist. If a device has only 6 EPWMs (PWM_CH is 7),
// the array will include pointers for up to 6 EPWM register structures plus one
// dummy pointer in the ePWM[0] location.
volatile struct EPWM_REGS *ePWM[PWM_CH] {&EPwm1Regs, &EPwm1Regs, &EPwm2Regs,
&EPwm3Regs, &EPwm4Regs, &EPwm5Regs, &EPwm6Regs, &EPwm7Regs,
&EPwm8Regs, &EPwm9Regs, &EPwm10Regs, &EPwm11Regs, &EPwm12Regs,
&EPwm13Regs, &EPwm14Regs, &EPwm15Regs, &EPwm16Regs};
```

Step 4. MEP_ScaleFactor Initialization

After power up, the SFO_MepEn_V5(n) function needs a typical scale factor starting seed value in MEP_ScaleFactor[0]. This value can be conveniently determined using one of the ePWM modules to run the SFO_MepDis_V5(n) function prior to initializing the PWM settings for the application. The SFO_MepDis_V5(n) function does not require a starting scale factor value.

As part of the one-time initialization code, include the following:

Example 15-10. Initialization With a Scale Factor Value

```
// MEP_ScaleFactor variables initialized using function SFO_MepDis_V5
Uint16 I;
for (I=1; i<PWM_CH; I++) // For channels 1-16
{
    while (SFO_MepDis_V5(I)==0); // Calls MepDis until MEP_ScaleFactor updated
}

// Initialize MEP_ScaleFactor[0] with a typical MEP seed value
// required for SFO_MepEn_V5
MEP_ScaleFactor[0] = MEP_ScaleFactor[1];
```

Step 5. Application Code

While the application is running, fluctuations in both device temperature and supply voltage may be expected. To be sure that optimal scale factors are used for each ePWM module, the SFO function should be re-run periodically as part of a slower background loop. Some examples of this are shown here.

Example 15-11. SFO Function Calls

```

main()
{
    Uint16 current_ch = 1; // keeps track of current HRPWM channel being calibrated
    Uint16 status;

    // User code

    // Case 1: All ePWMs are running in HRPWM mode
    //         Here, the minimum duty cycle limitation is 6 clock cycles.

        status = SFO_MepEn_V5(current_ch); // MepEn called here
        if (status ==1)                    // if MEP_ScaleFactor has been updated
        {
            current_ch++;                  // move on to the next channel
        }

        else if (status==2)                // if MEP_ScaleFactor differs from
        {                                   // MEP_ScaleFactor[0] seed by more than
            error();                        // +/-15, flag an error
        }
        if (current_ch==PWM_CH)           // if last channel has been reached
        {
            current_ch = 1;                 // go back to channel 1
        }

    // Case 2: all ePWMs except one are running in HRPWM mode.
    //         One of the ePWM channels (ePWM16 in this example) is used
    //         for SFO_MepDis_V5 scale factor calibration.
    //         Here, the minimum duty cycle limitation is 3 clock cycles.

    //         HRPWM16 diagnostics circuitry is used to estimate the MEP steps
    //         with the assumption that all HRPWM channels behave similarly
    //         though they may not be identical.

    while (SFO_MepDis_V5(16)==0); // wait until MEP_ScaleFactor[16] updated
    for (I=1; I<(PWM_CH-1); I++)    // Update scale factors for ePWM 1-15.
    {
        MEP_ScaleFactor[i] = MEP_ScaleFactor[16];
    }
}

```

15.2.2.11 ePWM Behavior During Emulation

To configure the ePWM to stop during emulation suspend events (for example, debugger breakpoints), set up the ePWM and the Debug Subsystem:

1. Set TBCTL.FREE_SOFT= 0 or 1 (see register description for more details). This will allow the Suspend_Control signal from the Debug Subsystem ([Chapter 27](#)) to stop and start the ePWM. Note that if FREE_SOFT = 2 or 3, the Suspend_Control signal is ignored and the ePWM is free running regardless of any debug suspend event. This FREE_SOFT bit gives local control from a module perspective to gate the suspend signal coming from the Debug Subsystem.
2. Set the appropriate xxx_Suspend_Control register = 0x9, as described in [Section 27.1.1.1, Debug Suspend Support for Peripherals](#). Choose the register appropriate to the peripheral you want to suspend during a suspend event.

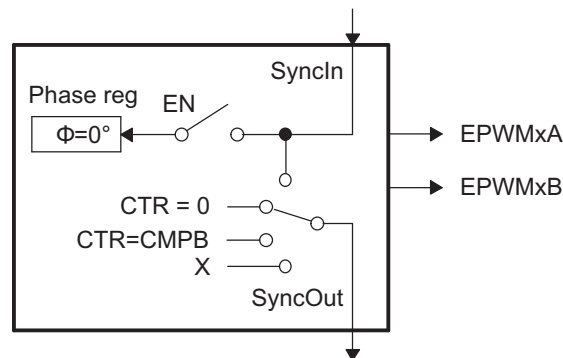
15.2.3 Use Cases

An ePWM module has all the local resources necessary to operate completely as a standalone module or to operate in synchronization with other identical ePWM modules.

15.2.3.1 Overview of Multiple Modules

Previously in this user's guide, all discussions have described the operation of a single module. To facilitate the understanding of multiple modules working together in a system, the ePWM module described in reference is represented by the more simplified block diagram shown in [Figure 15-55](#). This simplified ePWM block shows only the key resources needed to explain how a multiswitch power topology is controlled with multiple ePWM modules working together.

Figure 15-55. Simplified ePWM Module



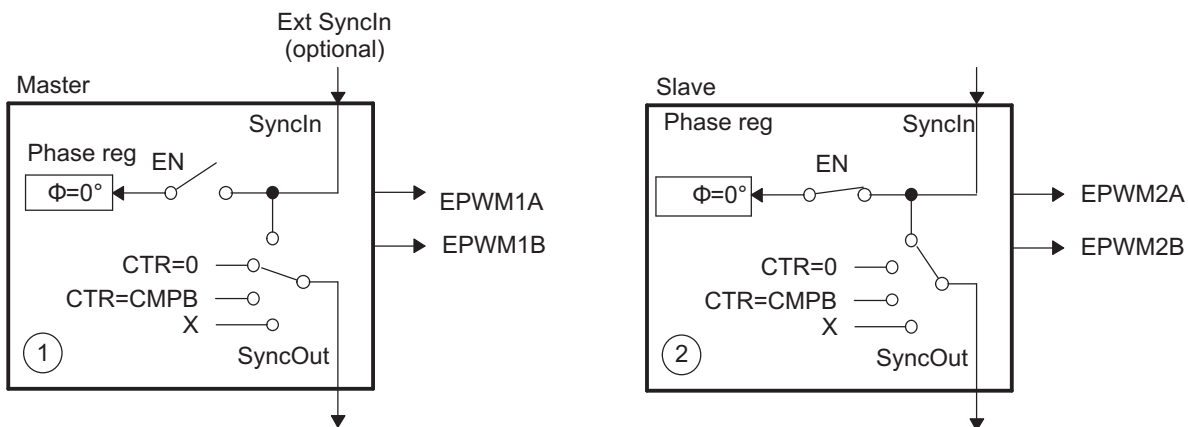
15.2.3.2 Key Configuration Capabilities

The key configuration choices available to each module are as follows:

- Options for SyncIn
 - Load own counter with phase register on an incoming sync strobe—enable (EN) switch closed
 - Do nothing or ignore incoming sync strobe—enable switch open
 - Sync flow-through - SyncOut connected to SyncIn
 - Master mode, provides a sync at PWM boundaries—SyncOut connected to CTR = PRD
 - Master mode, provides a sync at any programmable point in time—SyncOut connected to CTR = CMPB
 - Module is in standalone mode and provides No sync to other modules—SyncOut connected to X (disabled)
- Options for SyncOut
 - Sync flow-through - SyncOut connected to SyncIn
 - Master mode, provides a sync at PWM boundaries—SyncOut connected to CTR = PRD
 - Master mode, provides a sync at any programmable point in time—SyncOut connected to CTR = CMPB
 - Module is in standalone mode and provides No sync to other modules—SyncOut connected to X (disabled)

For each choice of SyncOut, a module may also choose to load its own counter with a new phase value on a SyncIn strobe input or choose to ignore it, i.e., via the enable switch. Although various combinations are possible, the two most common—master module and slave module modes—are shown in [Figure 15-56](#).

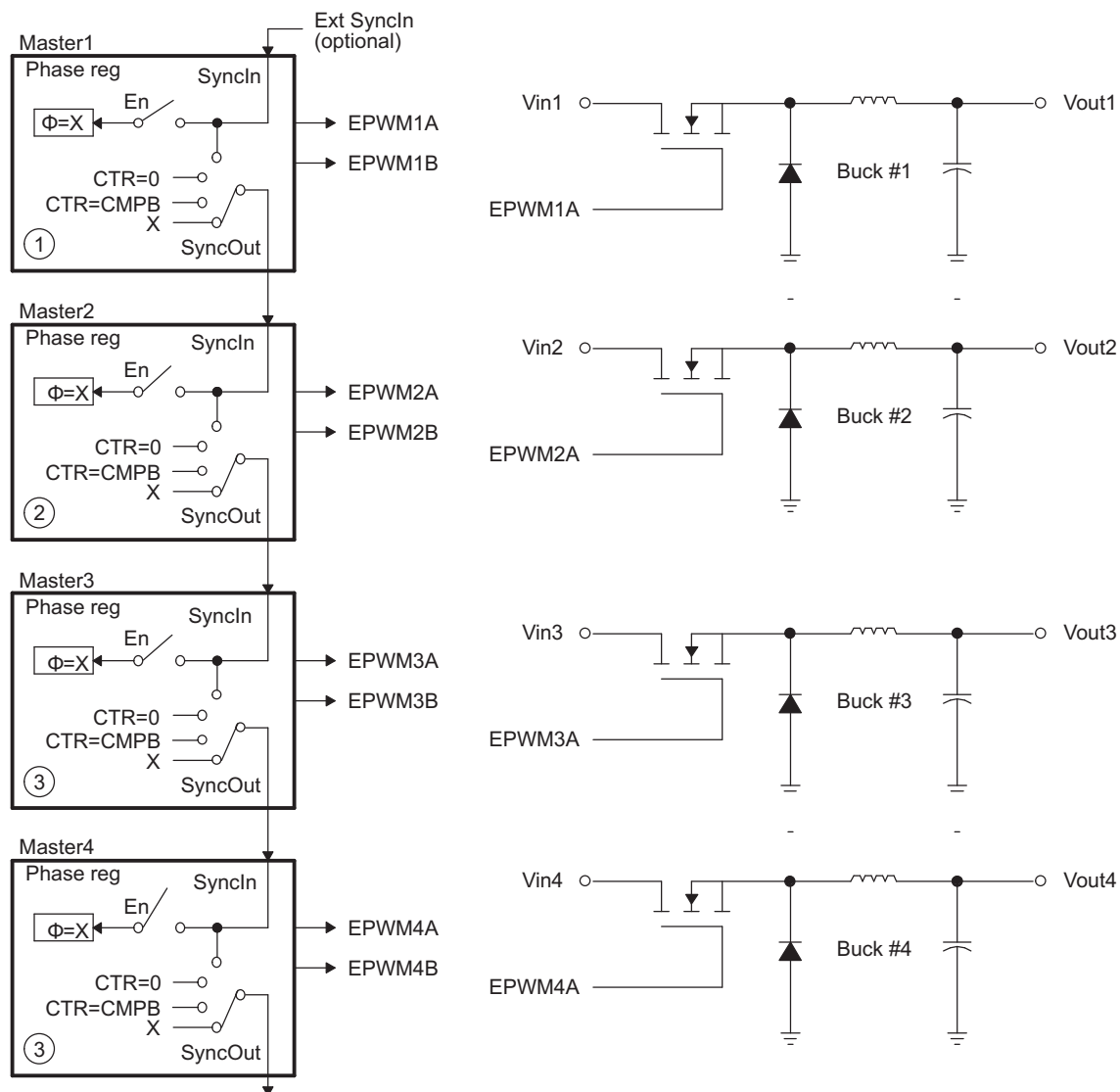
Figure 15-56. EPWM1 Configured as a Typical Master, EPWM2 Configured as a Slave



15.2.3.3 Controlling Multiple Buck Converters With Independent Frequencies

One of the simplest power converter topologies is the buck. A single ePWM module configured as a master can control two buck stages with the same PWM frequency. If independent frequency control is required for each buck converter, then one ePWM module must be allocated for each converter stage. Figure 15-57 shows four buck stages, each running at independent frequencies. In this case, all four ePWM modules are configured as Masters and no synchronization is used. Figure 15-58 shows the waveforms generated by the setup shown in Figure 15-57; note that only three waveforms are shown, although there are four stages.

Figure 15-57. Control of Four Buck Stages. Here $F_{PWM1} \neq F_{PWM2} \neq F_{PWM3} \neq F_{PWM4}$



NOTE: $\Phi = X$ indicates value in phase register is a "don't care"

Figure 15-58. Buck Waveforms for Figure 15-57 (Note: Only three bucks shown here)

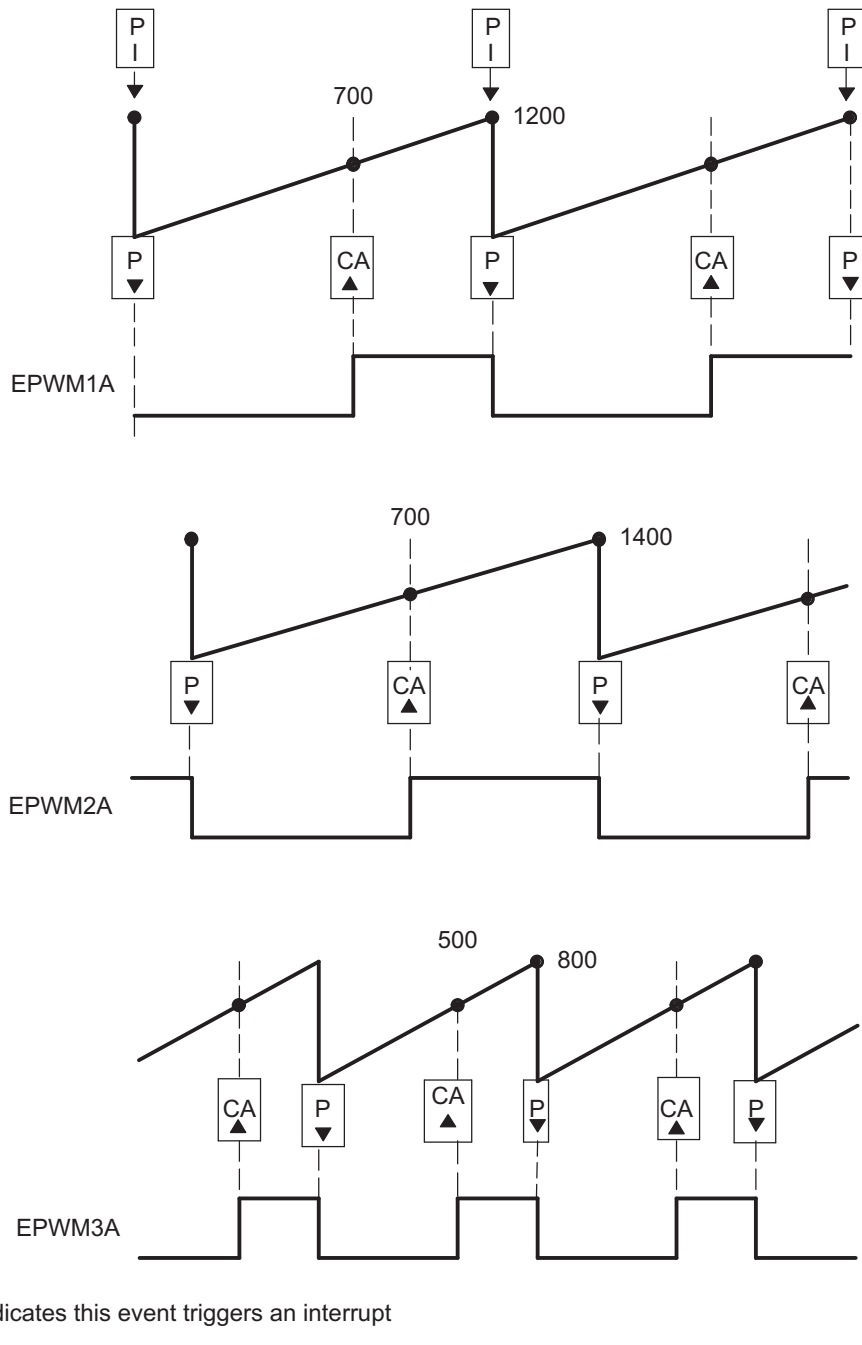


Table 15-48. EPWM1 Initialization for Figure 15-58

Register	Bit	Value	Comments
TBPRD	TBPRD	1200 (4B0h)	Period = 1201 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	PRD	AQ_CLEAR	
	CAU	AQ_SET	

Table 15-49. EPWM2 Initialization for Figure 15-58

Register	Bit	Value	Comments
TBPRD	TBPRD	1400 (578h)	Period = 1401 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	PRD	AQ_CLEAR	
	CAU	AQ_SET	

Table 15-50. EPWM3 Initialization for Figure 15-58

Register	Bit	Value	Comments
TBPRD	TBPRD	800 (320h)	Period = 801 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_DISABLE	
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	PRD	AQ_CLEAR	
	CAU	AQ_SET	

Example 15-12. Configuration for Example in Figure 15-58

```
// Run Time (Note: Example execution of one run-time instance)
//=====
EPwm1Regs.CMPA.half.CMPA = 700;           // adjust duty for output EPWM1A
EPwm2Regs.CMPA.half.CMPA = 700;           // adjust duty for output EPWM2A
EPwm3Regs.CMPA.half.CMPA = 500;           // adjust duty for output EPWM3A
```

15.2.3.4 Controlling Multiple Buck Converters With Same Frequencies

If synchronization is a requirement, ePWM module 2 can be configured as a slave and can operate at integer multiple (N) frequencies of module 1. The sync signal from master to slave ensures these modules remain locked. Figure 15-59 shows such a configuration; Figure 15-60 shows the waveforms generated by the configuration.

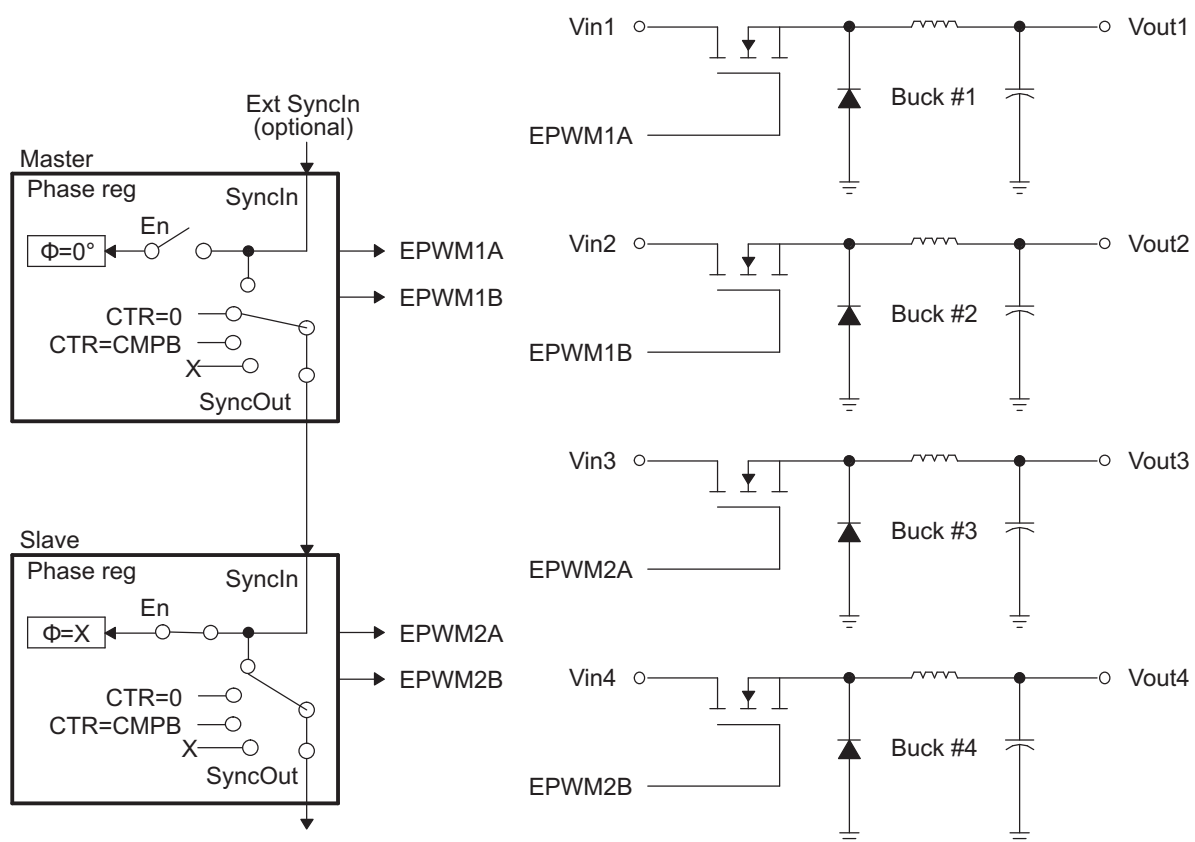
Figure 15-59. Control of Four Buck Stages. (Note: $F_{PWM2} = N \times F_{PWM1}$)


Figure 15-60. Buck Waveforms for Figure 15-59 (Note: $F_{PWM2} = F_{PWM1}$)

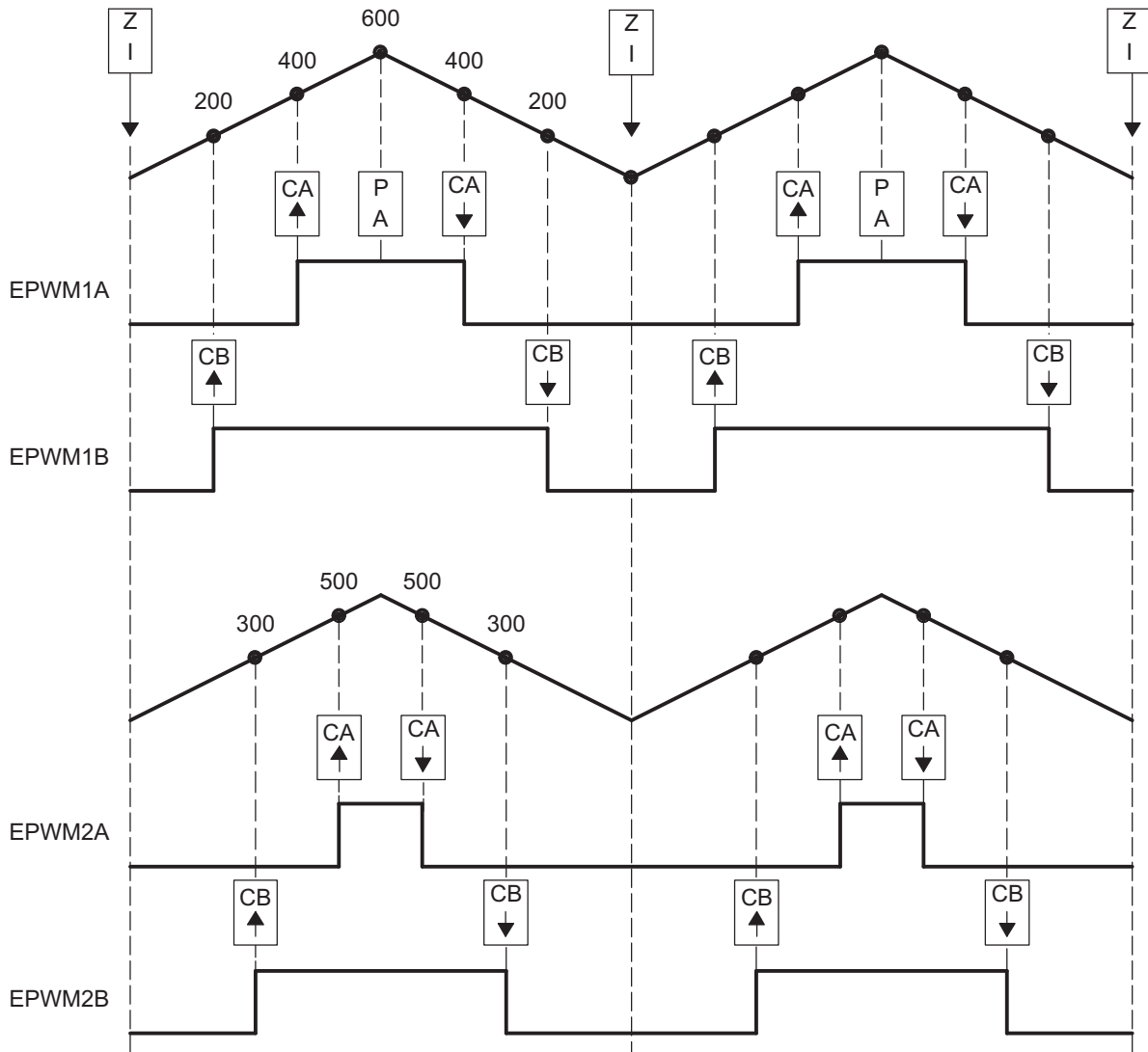


Table 15-51. EPWM1 Initialization for Figure 15-59

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 1200 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_CTR_ZERO	Sync down-stream module
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM1A
	CAD	AQ_CLEAR	
AQCTLB	CBU	AQ_SET	Set actions for EPWM1B
	CBD	AQ_CLEAR	

Table 15-52. EPWM2 Initialization for Figure 15-59

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 1200 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Phase loading enabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM2A
	CAD	AQ_CLEAR	
AQCTLB	CBU	AQ_SET	Set actions for EPWM2B
	CBD	AQ_CLEAR	

Example 15-13. Code Snippet for Configuration in Figure 15-59

```
// Run Time (Note: Example execution of one run-time instance)
//=====
EPwm1Regs.CMPA.half.CMPA = 400;    // adjust duty for output EPWM1A
EPwm1Regs.CMPB = 200;             // adjust duty for output EPWM1B
EPwm2Regs.CMPA.half.CMPA = 500;   // adjust duty for output EPWM2A
EPwm2Regs.CMPB = 300;             // adjust duty for output EPWM2B
```

15.2.3.5 Controlling Multiple Half H-Bridge (HHB) Converters

Topologies that require control of multiple switching elements can also be addressed with these same ePWM modules. It is possible to control a Half-H bridge stage with a single ePWM module. This control can be extended to multiple stages. Figure 15-61 shows control of two synchronized Half-H bridge stages where stage 2 can operate at integer multiple (N) frequencies of stage 1. Figure 15-62 shows the waveforms generated by the configuration shown in Figure 15-61.

Module 2 (slave) is configured for Sync flow-through; if required, this configuration allows for a third Half-H bridge to be controlled by PWM module 3 and also, most importantly, to remain in synchronization with master module 1.

Figure 15-61. Control of Two Half-H Bridge Stages ($F_{P_{WM2}} = N \times F_{P_{WM1}}$)

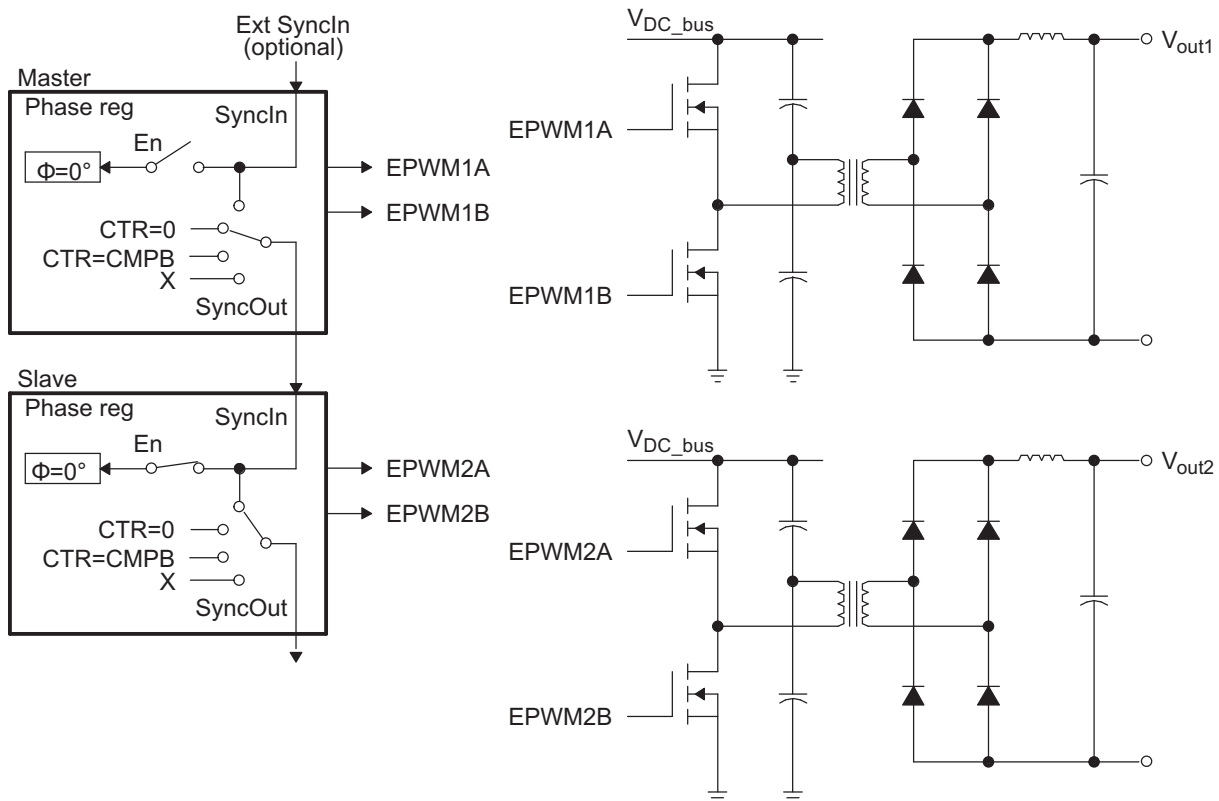


Figure 15-62. Half-H Bridge Waveforms for Figure 15-61 (Note: Here $F_{PWM2} = F_{PWM1}$)

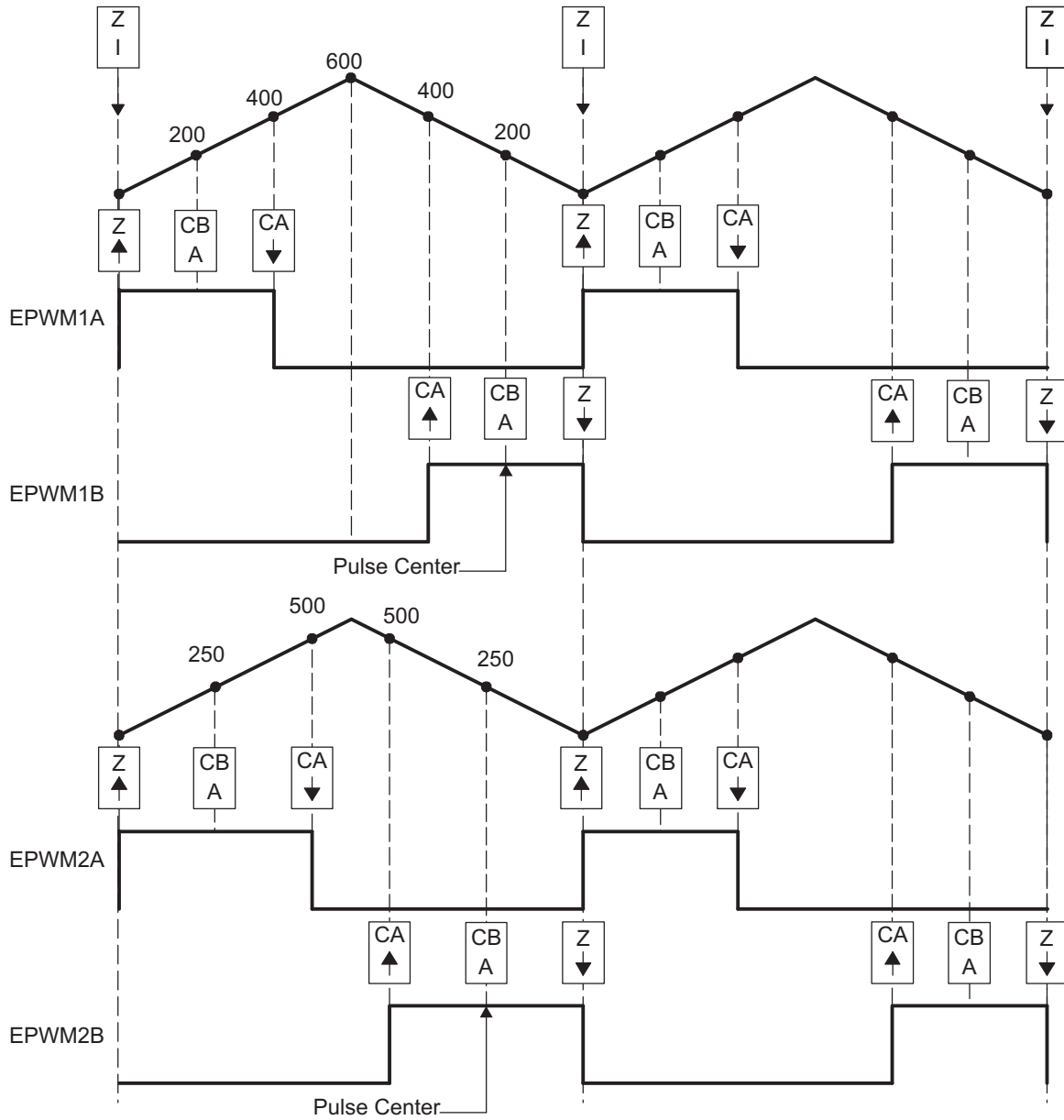


Table 15-53. EPWM1 Initialization for Figure 15-61

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 1200 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_CTR_ZERO	Sync down-stream module
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	ZRO	AQ_SET	Set actions for EPWM1A
	CAU	AQ_CLEAR	
AQCTLB	ZRO	AQ_CLEAR	Set actions for EPWM1B
	CAD	AQ_SET	

Table 15-54. EPWM2 Initialization for Figure 15-61

Register	Bit	Value	Comments
TBPRD	TBPRD	600 (258h)	Period = 1200 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Phase loading enabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	ZRO	AQ_SET	Set actions for EPWM2A
	CAU	AQ_CLEAR	
AQCTLB	ZRO	AQ_CLEAR	Set actions for EPWM2B
	CAD	AQ_SET	

Example 15-14. Code Snippet for Configuration in Figure 15-61

```
// Run Time (Note: Example execution of one run-time instance)
//=====
EPwm1Regs.CMPA.half.CMPA = 400; // adjust duty for output EPWM1A
EPwm1Regs.CMPB = 200;          // adjust duty for output EPWM1B
EPwm2Regs.CMPA.half.CMPA = 500; // adjust duty for output EPWM2A
EPwm2Regs.CMPB = 250;          // adjust duty for output EPWM2B
```


15.2.3.6 Controlling Dual 3-Phase Inverters for Motors (ACI and PMSM)

The idea of multiple modules controlling a single power stage can be extended to the 3-phase Inverter case. In such a case, six switching elements can be controlled using three PWM modules, one for each leg of the inverter. Each leg must switch at the same frequency and all legs must be synchronized. A master + two slaves configuration can easily address this requirement. Figure 15-63 shows how six PWM modules can control two independent 3-phase Inverters; each running a motor.

As in the cases shown in the previous sections, we have a choice of running each inverter at a different frequency (module 1 and module 4 are masters as in Figure 15-63), or both inverters can be synchronized by using one master (module 1) and five slaves. In this case, the frequency of modules 4, 5, and 6 (all equal) can be integer multiples of the frequency for modules 1, 2, 3 (also all equal).

Figure 15-63. Control of Dual 3-Phase Inverter Stages as Is Commonly Used in Motor Control

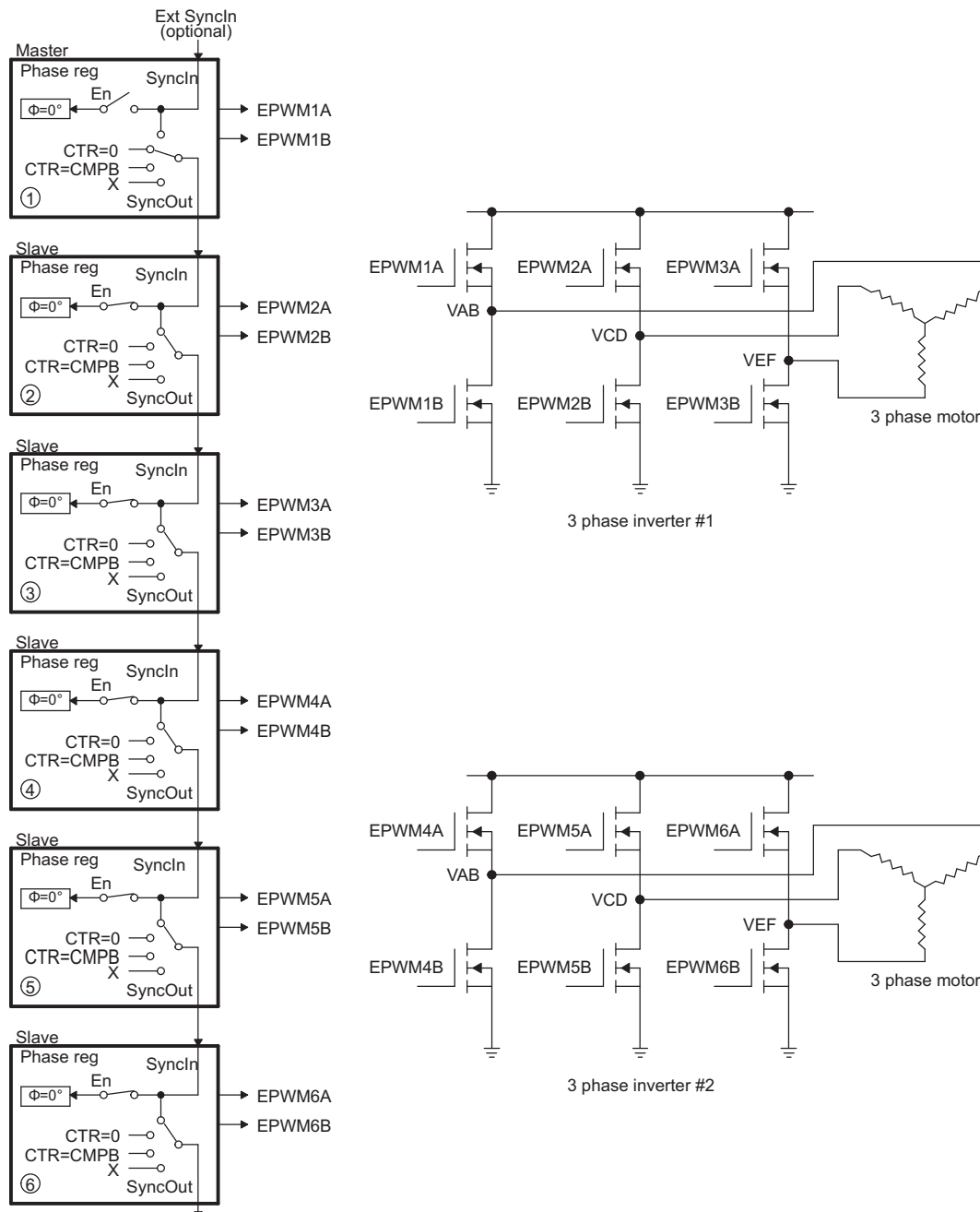


Figure 15-64. 3-Phase Inverter Waveforms for Figure 15-63 (Only One Inverter Shown)

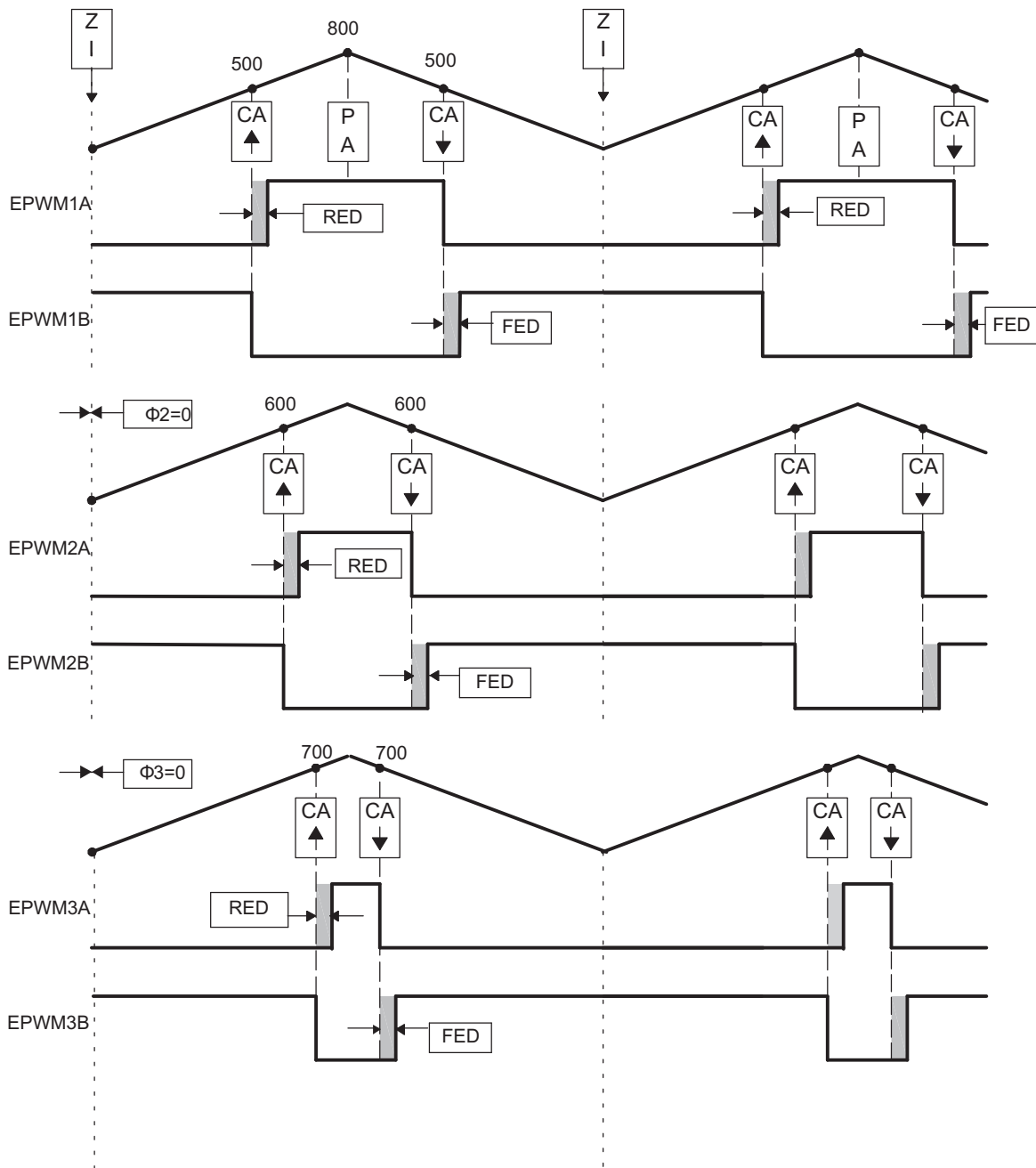


Table 15-55. EPWM1 Initialization for Figure 15-63

Register	Bit	Value	Comments
TBPRD	TBPRD	800 (320h)	Period = 1600 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_CTR_ZERO	Sync down-stream module
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM1A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	50	FED = 50 TBCLKs
	DBRED	50	RED = 50 TBCLKs

Table 15-56. EPWM2 Initialization for Figure 15-63

Register	Bit	Value	Comments
TBPRD	TBPRD	800 (320h)	Period = 1600 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Slave module
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM2A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	50	FED = 50 TBCLKs
	DBRED	50	RED = 50 TBCLKs

Table 15-57. EPWM3 Initialization for Figure 15-63

Register	Bit	Value	Comments
TBPRD	TBPRD	800 (320h)	Period = 1600 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Slave module
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM3A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	50	FED = 50 TBCLKs
	DBRED	50	RED = 50 TBCLKs

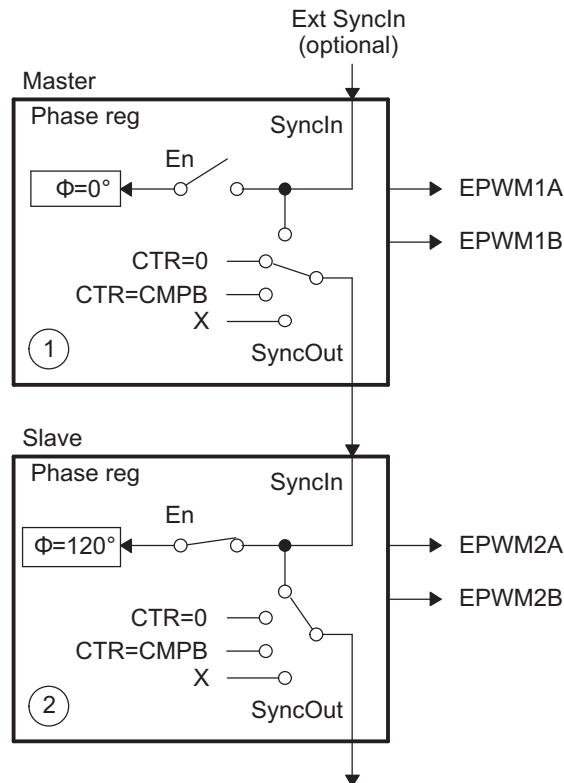
Example 15-15. Code Snippet for Configuration in Figure 15-63

```
// Run Time (Note: Example execution of one run-time instance)
//=====
EPwm1Regs.CMPA.half.CMPA = 500; // adjust duty for output EPWM1A
EPwm2Regs.CMPA.half.CMPA = 600; // adjust duty for output EPWM2A
EPwm3Regs.CMPA.half.CMPA = 700; // adjust duty for output EPWM3A
```

15.2.3.7 Practical Applications Using Phase Control Between PWM Modules

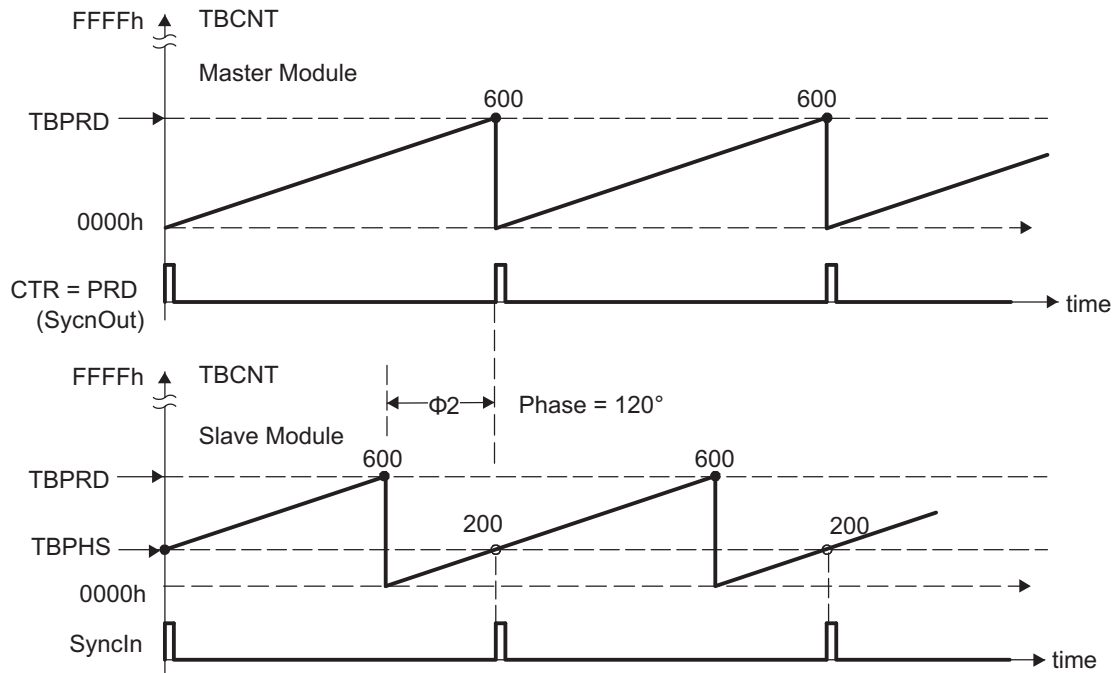
So far, none of the examples have made use of the phase register (TBPHS). It has either been set to zero or its value has been a don't care. However, by programming appropriate values into TBPHS, multiple PWM modules can address another class of power topologies that rely on phase relationship between legs (or stages) for correct operation. As described in the TB module section, a PWM module can be configured to allow a SyncIn pulse to cause the TBPHS register to be loaded into the TBCNT register. To illustrate this concept, [Figure 15-65](#) shows a master and slave module with a phase relationship of 120°, that is, the slave leads the master.

Figure 15-65. Configuring Two PWM Modules for Phase Control



[Figure 15-66](#) shows the associated timing waveforms for this configuration. Here, TBPRD = 600 for both master and slave. For the slave, TBPHS = 200 ($200/600 \times 360^\circ = 120^\circ$). Whenever the master generates a SyncIn pulse (CTR = PRD), the value of TBPHS = 200 is loaded into the slave TBCNT register so the slave time-base is always leading the master's time-base by 120°.

Figure 15-66. Timing Waveforms Associated With Phase Control Between 2 Modules



15.2.3.8 Controlling a 3-Phase Interleaved DC/DC Converter

A popular power topology that makes use of phase-offset between modules is shown in [Figure 15-67](#). This system uses three PWM modules, with module 1 configured as the master. To work, the phase relationship between adjacent modules must be $F = 120^\circ$. This is achieved by setting the slave TBPHS registers 2 and 3 with values of 1/3 and 2/3 of the period value, respectively. For example, if the period register is loaded with a value of 600 counts, then TBPHS (slave 2) = 200 and TBPHS (slave 3) = 400. Both slave modules are synchronized to the master 1 module.

This concept can be extended to four or more phases, by setting the TBPHS values appropriately. The following formula gives the TBPHS values for N phases:

$$TBPHS(N,M) = (TBPRD/N) \times (M - 1)$$

Where:

N = number of phases

M = PWM module number

For example, for the 3-phase case (N = 3), TBPRD = 600,

$$TBPHS(3,2) = (600/3) \times (2 - 1) = 200 \times 1 = 200 \text{ (Phase value for Slave module 2)}$$

$$TBPHS(3,3) = (600/3) \times (3 - 1) = 200 \times 2 = 400 \text{ (Phase value for Slave module 3)}$$

[Figure 15-68](#) shows the waveforms for the configuration in [Figure 15-67](#).

Figure 15-67. Control of a 3-Phase Interleaved DC/DC Converter

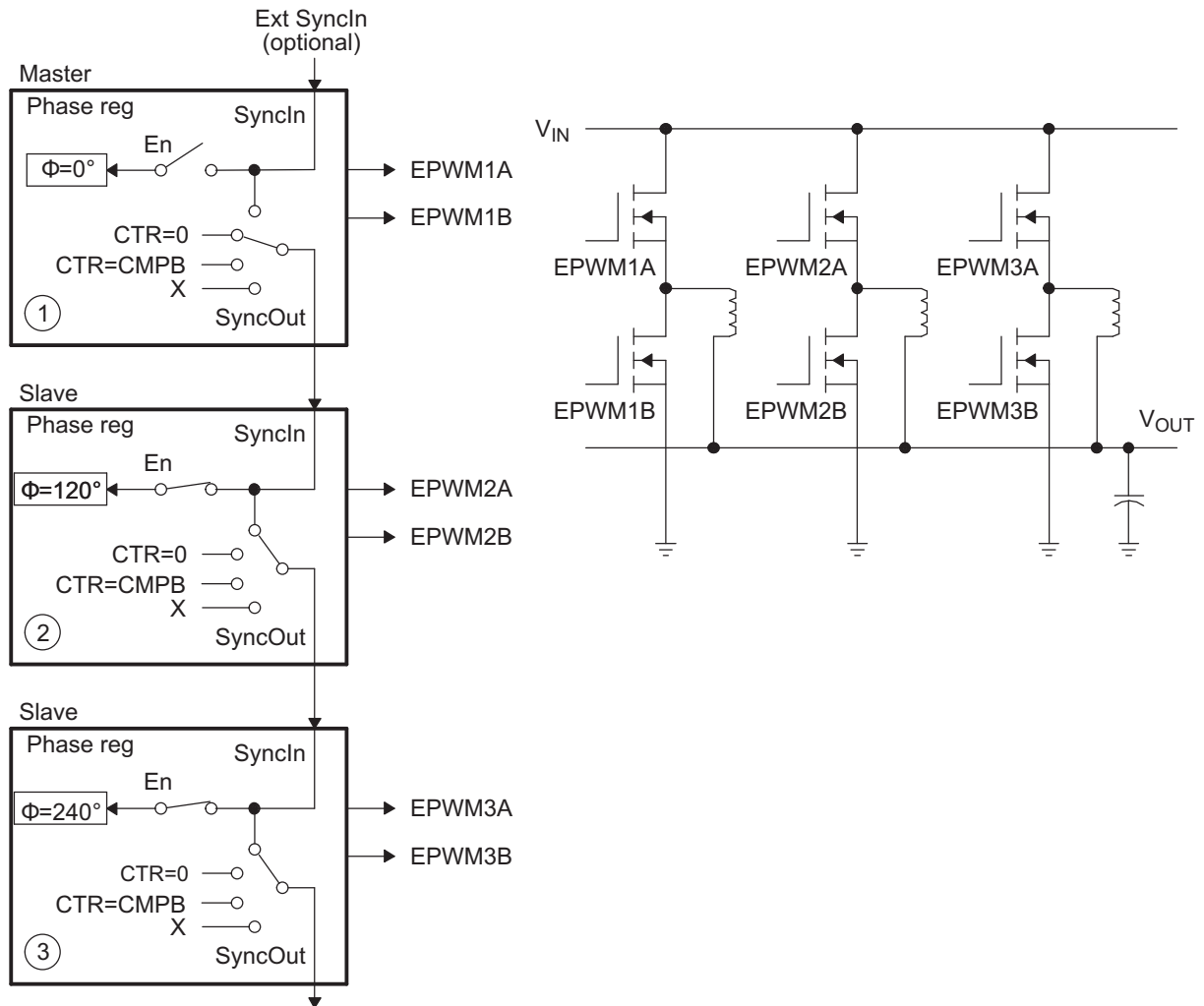


Figure 15-68. 3-Phase Interleaved DC/DC Converter Waveforms for Figure 15-67

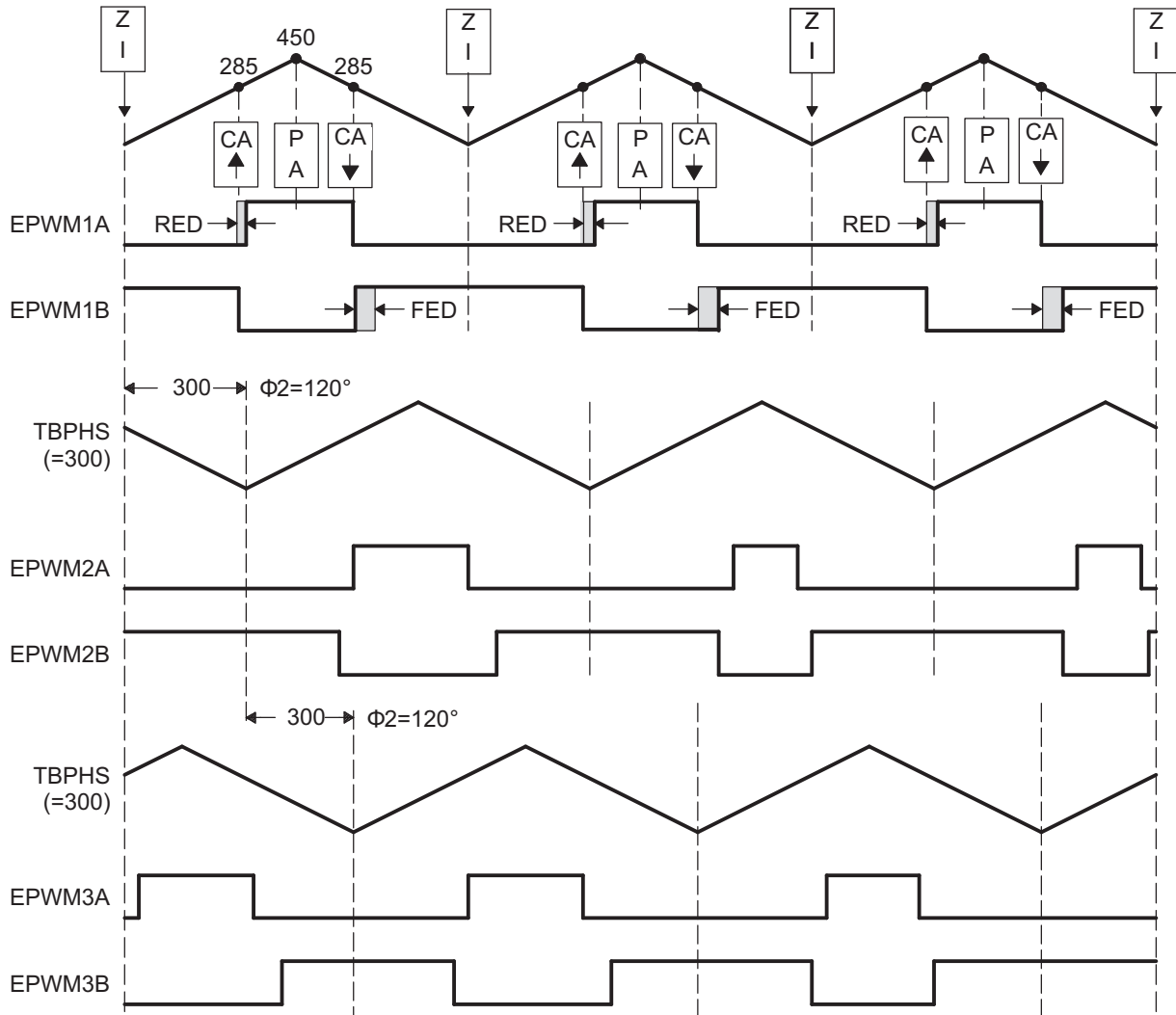


Table 15-58. EPWM1 Initialization for Figure 15-67

Register	Bit	Value	Comments
TBPRD	TBPRD	450 (1C2h)	Period = 900 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_CTR_ZERO	Sync down-stream module
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM1A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	20	FED = 20 TBCLKs
	DBRED	20	RED = 20 TBCLKs

Table 15-59. EPWM2 Initialization for Figure 15-67

Register	Bit	Value	Comments
TBPRD	TBPRD	450 (1C2h)	Period = 900 TBCLK counts
TBPHS	TBPHS	300	Phase = $(300/900) \times 360 = 120^\circ$
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Slave module
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
	PHSDIR	TB_DOWN	Count DOWN on sync
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM2A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	20	FED = 20 TBCLKs
	DBRED	20	RED = 20 TBCLKs

Table 15-60. EPWM3 Initialization for Figure 15-67

Register	Bit	Value	Comments
TBPRD	TBPRD	450 (1C2h)	Period = 900 TBCLK counts
TBPHS	TBPHS	300	Phase = $(300/900) \times 360 = 120^\circ$
TBCTL	CTRMODE	TB_UPDOWN	
	PHSEN	TB_ENABLE	Slave module
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
	PHSDIR	TB_UP	Count UP on sync
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	CAU	AQ_SET	Set actions for EPWM3A
	CAD	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	20	FED = 20 TBCLKs
	DBRED	20	RED = 20 TBCLKs

Example 15-16. Code Snippet for Configuration in Figure 15-67

```
// Run Time (Note: Example execution of one run-time instance)
//=====
EPwm1Regs.CMPA.half.CMPA = 285;           // adjust duty for output EPWM1A
EPwm2Regs.CMPA.half.CMPA = 285;           // adjust duty for output EPWM2A
EPwm3Regs.CMPA.half.CMPA = 285;           // adjust duty for output EPWM3A
```

15.2.3.9 Controlling Zero Voltage Switched Full Bridge (ZVSFB) Converter

The example given in [Figure 15-69](#) assumes a static or constant phase relationship between legs (modules). In such a case, control is achieved by modulating the duty cycle. It is also possible to dynamically change the phase value on a cycle-by-cycle basis. This feature lends itself to controlling a class of power topologies known as *phase-shifted full bridge*, or *zero voltage switched full bridge*. Here the controlled parameter is not duty cycle (this is kept constant at approximately 50 percent); instead it is the phase relationship between legs. Such a system can be implemented by allocating the resources of two PWM modules to control a single power stage, which in turn requires control of four switching elements. [Figure 15-70](#) shows a master/slave module combination synchronized together to control a full H-bridge. In this case, both master and slave modules are required to switch at the same PWM frequency. The phase is controlled by using the slave's phase register (TBPHS). The master's phase register is not used and therefore can be initialized to zero.

Figure 15-69. Controlling a Full-H Bridge Stage ($F_{PWM2} = F_{PWM1}$)

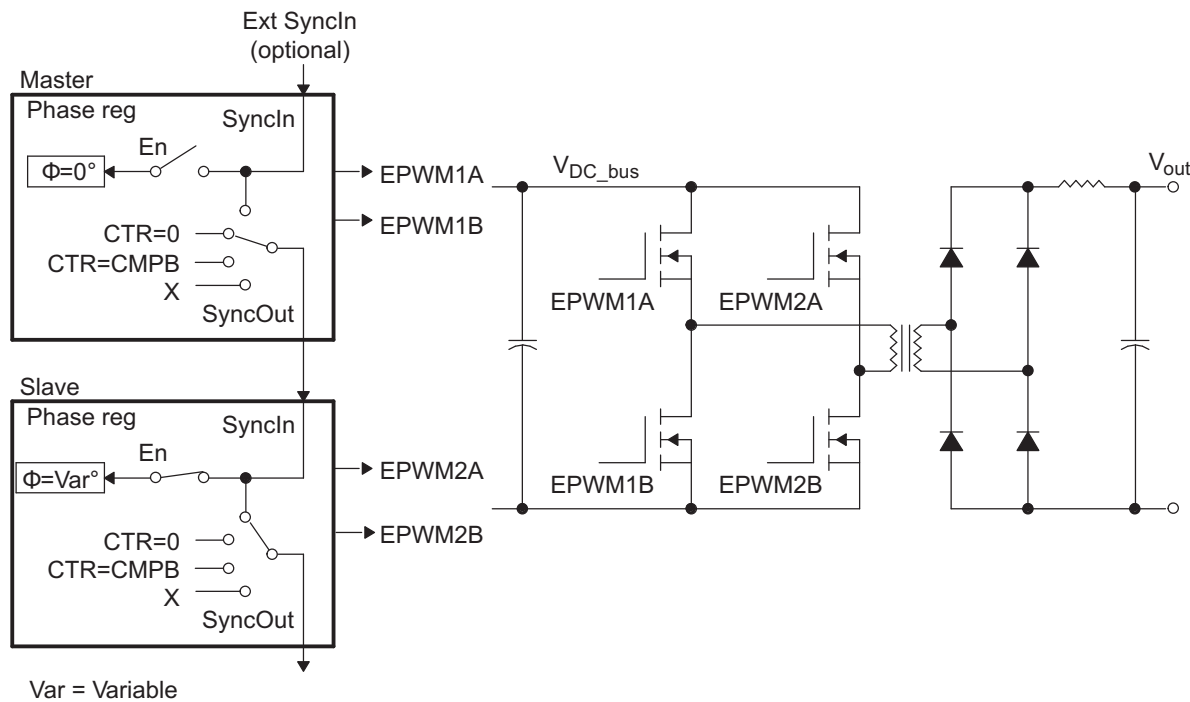


Figure 15-70. ZVS Full-H Bridge Waveforms

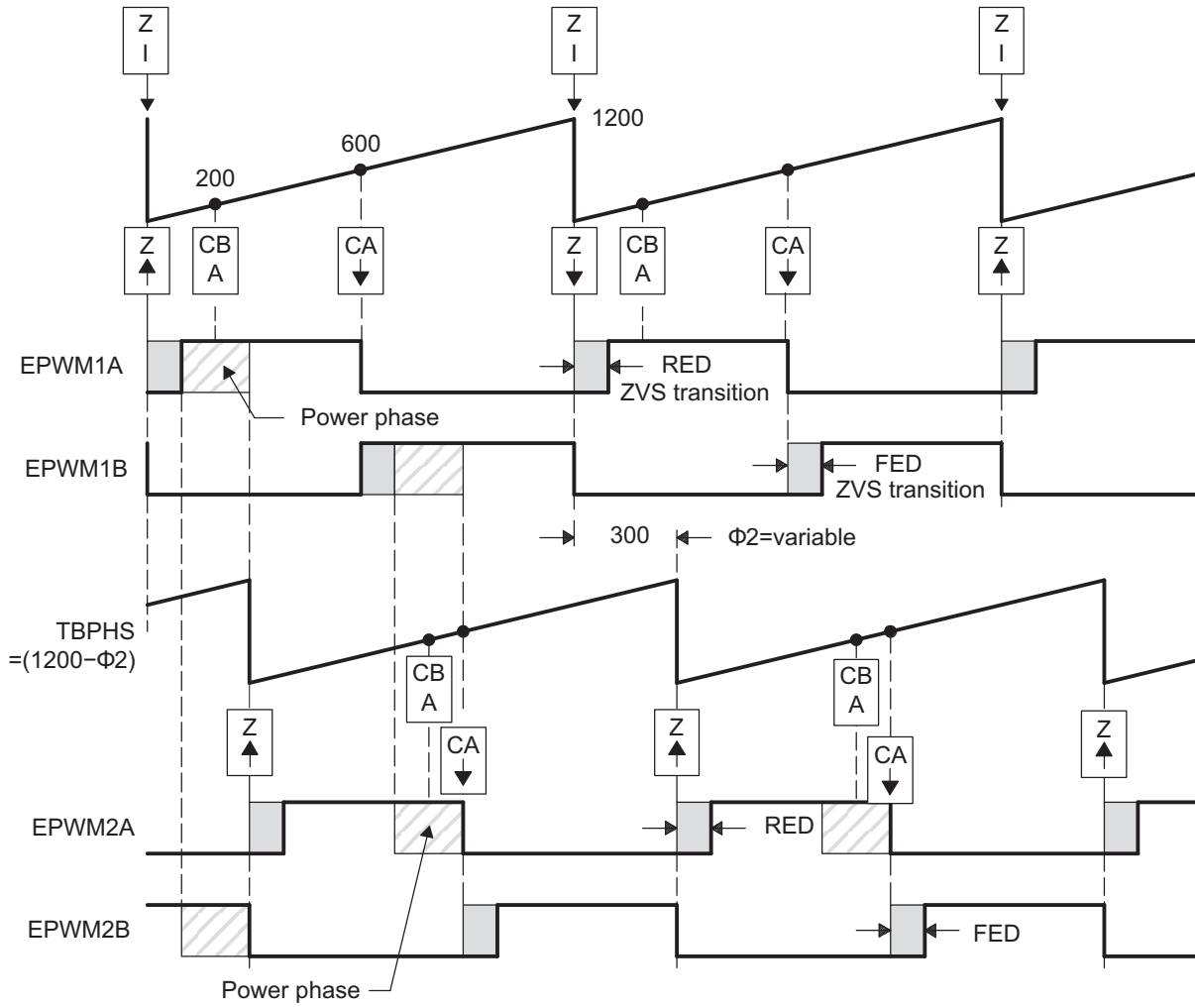


Table 15-61. EPWM1 Initialization for Figure 15-69

Register	Bit	Value	Comments
TBPRD	TBPRD	1200 (4B0h)	Period = 1201 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_DISABLE	Phase loading disabled
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_CTR_ZERO	Sync down-stream module
CMPA	CMPA	600 (258h)	Set 50% duty for EPWM1A
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	ZRO	AQ_SET	Set actions for EPWM1A
	CAU	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	50	FED = 50 TBCLKs
	DBRED	70	RED = 70 TBCLKs

Table 15-62. EPWM2 Initialization for Figure 15-69

Register	Bit	Value	Comments
TBPRD	TBPRD	1200 (4B0h)	Period = 1201 TBCLK counts
TBPHS	TBPHS	0	Clear Phase Register to 0
TBCTL	CTRMODE	TB_UP	
	PHSEN	TB_ENABLE	Slave module
	PRDL	TB_SHADOW	
	SYNCOSEL	TB_SYNC_IN	Sync flow-through
CMPA	CMPA	600 (258h)	Set 50% duty for EPWM2A
CMPCTL	SHDWAMODE	CC_SHADOW	
	SHDWBMODE	CC_SHADOW	
	LOADAMODE	CC_CTR_ZERO	Load on CTR = 0
	LOADBMODE	CC_CTR_ZERO	Load on CTR = 0
AQCTLA	ZRO	AQ_SET	Set actions for EPWM2A
	CAU	AQ_CLEAR	
DBCTL	MODE	DB_FULL_ENABLE	Enable Dead-band module
	POLSEL	DB_ACTV_HIC	Active Hi complementary
DBFED	DBFED	30	FED = 30 TBCLKs
	DBRED	40	RED = 40 TBCLKs

Example 15-17. Code Snippet for Configuration in Figure 15-69

```
// Run Time (Note: Example execution of one run-time instance)
//=====
EPwm2Regs.TBPHS = 1200-300; // Set Phase reg to 300/1200 * 360 = 90 deg
EPwm1Regs.DBFED = FED1_NewValue; // Update ZVS transition interval
EPwm1Regs.DBRED = RED1_NewValue; // Update ZVS transition interval
EPwm2Regs.DBFED = FED2_NewValue; // Update ZVS transition interval
```

Example 15-17. Code Snippet for Configuration in Figure 15-69 (continued)

```
EPwm2Regs.DBRED = RED2_NewValue;           // Update ZVS transition interval
```

15.2.4 EPWM Registers

Table 15-63 lists the memory-mapped registers for the EPWM. All register offset addresses not listed in Table 15-63 should be considered as reserved locations and the register contents should not be modified.

Table 15-63. EPWM Registers

Offset	Acronym	Register Name	Section
0h	TBCTL	Time-Base Control Register	Section 15.2.4.1
2h	TBSTS	Time-Base Status Register	Section 15.2.4.2
4h	TBPHSHR	Extension for HRPWM Phase Register	Section 15.2.4.3
6h	TBPHS	Time-Base Phase Register	Section 15.2.4.4
8h	TBCNT	Time-Base Counter Register	Section 15.2.4.5
Ah	TBPRD	Time-Base Period Register	Section 15.2.4.6
Eh	CMPCTL	Counter-Compare Control Register	Section 15.2.4.7
10h	CMPAHR	Extension for HRPWM Counter-Compare A Register	Section 15.2.4.8
12h	CMPA	Counter-Compare A Register	Section 15.2.4.9
14h	CMPB	Counter-Compare B Register	Section 15.2.4.10
16h	AQCTLA	Action-Qualifier Control Register for Output A (EPWMxA)	Section 15.2.4.11
18h	AQCTLB	Action-Qualifier Control Register for Output B (EPWMxB)	Section 15.2.4.12
1Ah	AQSFR	Action-Qualifier Software Force Register	Section 15.2.4.13
1Ch	AQCSFR	Action-Qualifier Continuous S/W Force Register Set	Section 15.2.4.14
1Eh	DBCTL	Dead-Band Generator Control Register	Section 15.2.4.15
20h	DBRED	Dead-Band Generator Rising Edge Delay Count Register	Section 15.2.4.16
22h	DBFED	Dead-Band Generator Falling Edge Delay Count Register	Section 15.2.4.17
24h	TZSEL	Trip-Zone Select Register	Section 15.2.4.18
28h	TZCTL	Trip-Zone Control Register	Section 15.2.4.19
2Ah	TZEINT	Trip-Zone Enable Interrupt Register	Section 15.2.4.20
2Ch	TZFLG	Trip-Zone Flag Register	Section 15.2.4.21
2Eh	TZCLR	Trip-Zone Clear Register	Section 15.2.4.22
30h	TZFRC	Trip-Zone Force Register	Section 15.2.4.23
32h	ETSEL	Event-Trigger Selection Register	Section 15.2.4.24
34h	ETPS	Event-Trigger Pre-Scale Register	Section 15.2.4.25
36h	ETFLG	Event-Trigger Flag Register	Section 15.2.4.26
38h	ETCLR	Event-Trigger Clear Register	Section 15.2.4.27
3Ah	ETFRC	Event-Trigger Force Register	Section 15.2.4.28
3Ch	PCCTL	PWM-Chopper Control Register	Section 15.2.4.29
C0h	HRCNFG	HRPWM configuration register (HRCNFG)	Section 15.2.4.30

15.2.4.1 TBCTL Register (offset = 0h) [reset = 0h]

 TBCTL is shown in [Figure 15-71](#) and described in [Table 15-64](#).

Figure 15-71. TBCTL Register

15	14	13	12	11	10	9	8
FREE_SOFT		PHSDIR	CLKDIV			HSPCLKDIV	
R/W-0h		R/W-0h	R/W-0h			R/W-0h	
7	6	5	4	3	2	1	0
HSPCLKDIV		SWFSYNC	SYNCOSEL		PRDLD	PHSEN	CTRMODE
R/W-0h		R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-64. TBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation Mode Bits. These bits select the behavior of the ePWM time-base counter during emulation suspend events. Emulation debug events can be set up in the Debug Subsystem. 0h (R/W) = Stop after the next time-base counter increment or decrement 1h (R/W) = Stop when counter completes a whole cycle. (a) Up-count mode - stop when the time-base counter = period (TBCNT = TBPRD). (b) Down-count mode - stop when the time-base counter = 0000 (TBCNT = 0000h). (c) Up-down-count mode - stop when the time-base counter = 0000 (TBCNT = 0000h). 2h (R/W) = Free run 3h (R/W) = Free run
13	PHSDIR	R/W	0h	Phase Direction Bit. This bit is only used when the time-base counter is configured in the up-down-count mode. The PHSDIR bit indicates the direction the time-base counter (TBCNT) will count after a synchronization event occurs and a new phase value is loaded from the phase (TBPHS) register. This is irrespective of the direction of the counter before the synchronization event. In the up-count and down-count modes this bit is ignored. 0h (R/W) = Count down after the synchronization event. 1h (R/W) = Count up after the synchronization event.
12-10	CLKDIV	R/W	0h	Time-base Clock Prescale Bits. These bits determine part of the time-base clock prescale value. $TBCLK = SYSCLKOUT / (HSPCLKDIV * CLKDIV)$ 0h (R/W) = /1 (default on reset) 1h (R/W) = /2 2h (R/W) = /4 3h (R/W) = /8 4h (R/W) = /16 5h (R/W) = /32 6h (R/W) = /64 7h (R/W) = /128

Table 15-64. TBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-7	HSPCLKDIV	R/W	0h	<p>High-Speed Time-base Clock Prescale Bits. These bits determine part of the time-base clock prescale value. $TBCLK = SYSCLKOUT / (HSPCLKDIV * CLKDIV)$. This divisor emulates the HSPCLK in the TMS320x281x system as used on the Event Manager (EV) peripheral.</p> <p>0h (R/W) = /1 1h (R/W) = /2 (default on reset) 2h (R/W) = /4 3h (R/W) = /6 4h (R/W) = /8 5h (R/W) = /10 6h (R/W) = /12 7h (R/W) = /14</p>
6	SWFSYNC	R/W	0h	<p>Software Forced Synchronization Pulse.</p> <p>0h (R/W) = Writing a 0 has no effect and reads always return a 0. 1h (R/W) = Writing a 1 forces a one-time synchronization pulse to be generated. This event is ORed with the EPWMxSYNCl input of the ePWM module. SWFSYNC is valid (operates) only when EPWMxSYNCl is selected by SYNCOSSEL = 00.</p>
5-4	SYNCOSSEL	R/W	0h	<p>Synchronization Output Select. These bits select the source of the EPWMxSYNCO signal.</p> <p>0h (R/W) = EPWMxSYNCO: 1h (R/W) = CTR = 0 - Time-base counter equal to zero (TBCNT = 0000h) 2h (R/W) = CTR = CMPB - Time-base counter equal to counter-compare B (TBCNT = CMPB) 3h (R/W) = Disable EPWMxSYNCO signal</p>
3	PRDL	R/W	0h	<p>Active Period Register Load From Shadow Register Select</p> <p>0h (R/W) = The period register (TBPRD) is loaded from its shadow register when the time-base counter, TBCNT, is equal to zero. A write or read to the TBPRD register accesses the shadow register. 1h (R/W) = Load the TBPRD register immediately without using a shadow register. A write or read to the TBPRD register directly accesses the active register.</p>
2	PHSEN	R/W	0h	<p>Counter Register Load From Phase Register Enable</p> <p>0h (R/W) = Do not load the time-base counter (TBCNT) from the time-base phase register (TBPHS) 1h (R/W) = Load the time-base counter with the phase register when an EPWMxSYNCl input signal occurs or when a software synchronization is forced by the SWFSYNC bit.</p>
1-0	CTRM	R/W	0h	<p>Counter Mode. The time-base counter mode is normally configured once and not changed during normal operation. If you change the mode of the counter, the change will take effect at the next TBCLK edge and the current counter value shall increment or decrement from the value before the mode change. These bits set the time-base counter mode of operation as follows:</p> <p>0h (R/W) = Up-count mode 1h (R/W) = Down-count mode 2h (R/W) = Up-down-count mode 3h (R/W) = Stop-freeze counter operation (default on reset)</p>

15.2.4.2 TBSTS Register (offset = 2h) [reset = 0h]

 TBSTS is shown in [Figure 15-72](#) and described in [Table 15-65](#).

Figure 15-72. TBSTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					CTRMAX	SYNCI	CTRDIR
R-0h					0h	W1C-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-65. TBSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	
2	CTRMAX		0h	Time-Base Counter Max Latched Status Bit. 0h (R/W) = Reading a 0 indicates the time-base counter never reached its maximum value. Writing a 0 will have no effect. 1h (R/W) = Reading a 1 on this bit indicates that the time-base counter reached the max value 0xFFFF. Writing a 1 to this bit will clear the latched event.
1	SYNCI	W1C	0h	Input Synchronization Latched Status Bit. 0h (R/W) = Writing a 0 will have no effect. Reading a 0 indicates no external synchronization event has occurred. 1h (R/W) = Reading a 1 on this bit indicates that an external synchronization event has occurred (EPWMxSYNCI). Writing a 1 to this bit will clear the latched event.
0	CTRDIR	R	0h	Time-Base Counter Direction Status Bit. At reset, the counter is frozen, therefore, this bit has no meaning. To make this bit meaningful, you must first set the appropriate mode via TBCTL[CTRMODE]. 0h (R/W) = Time-Base Counter is currently counting down. 1h (R/W) = Time-Base Counter is currently counting up.

15.2.4.3 TBPHSHR Register (offset = 4h) [reset = 0h]

TBPHSHR is shown in [Figure 15-73](#) and described in [Table 15-66](#).

Figure 15-73. TBPHSHR Register

15	14	13	12	11	10	9	8
TBPHSH							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-66. TBPHSHR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	TBPHSH	R/W	0h	Time-base phase high-resolution bits
7-0	RESERVED	R	0h	

15.2.4.4 TBPHS Register (offset = 6h) [reset = 0h]

TBPHS is shown in [Figure 15-74](#) and described in [Table 15-67](#).

This register is only available on ePWM instances that include the high-resolution PWM (HRPWM) extension, otherwise, this location is reserved.

Figure 15-74. TBPHS Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBPHS															
R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-67. TBPHS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBPHS	R/W	0h	These bits set time-base counter phase of the selected ePWM relative to the time-base that is supplying the synchronization input signal. (a) If TBCTL[PHSEN] = 0, then the synchronization event is ignored and the time-base counter is not loaded with the phase. (b) If TBCTL[PHSEN] = 1, then the time-base counter (TBCNT) will be loaded with the phase (TBPHS) when a synchronization event occurs. The synchronization event can be initiated by the input synchronization signal (EPWMxSYNCl) or by a software forced synchronization.

15.2.4.5 TBCNT Register (offset = 8h) [reset = 0h]

TBCNT is shown in [Figure 15-75](#) and described in [Table 15-68](#).

Figure 15-75. TBCNT Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBCNT															
R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-68. TBCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBCNT	R/W	0h	Reading these bits gives the current time-base counter value. Writing to these bits sets the current time-base counter value. The update happens as soon as the write occurs. The write is NOT synchronized to the time-base clock (TBCLK) and the register is not shadowed.

15.2.4.6 TBPRD Register (offset = Ah) [reset = 0h]

TBPRD is shown in [Figure 15-76](#) and described in [Table 15-69](#).

Figure 15-76. TBPRD Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBPRD															
R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-69. TBPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBPRD	R/W	0h	These bits determine the period of the time-base counter. This sets the PWM frequency. Shadowing of this register is enabled and disabled by the TBCTL[PRDL] bit. By default this register is shadowed. (a) If TBCTL[PRDL] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the active register will be loaded from the shadow register when the time-base counter equals zero. (b) If TBCTL[PRDL] = 1, then the shadow is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. (c) The active and shadow registers share the same memory map address.

15.2.4.7 CMPCTL Register (offset = Eh) [reset = 0h]

CMPCTL is shown in [Figure 15-77](#) and described in [Table 15-70](#).

Figure 15-77. CMPCTL Register

15	14	13	12	11	10	9	8
RESERVED						SHDWBFULL	SHDWAFULL
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	SHDWBMODE	RESERVED	SHDWAMODE	LOADBMODE		LOADAMODE	
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h		R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-70. CMPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	
9	SHDWBFULL	R	0h	Counter-compare B (CMPB) Shadow Register Full Status Flag. This bit self clears once a load-strobe occurs. 0h (R/W) = CMPB shadow FIFO not full yet 1h (R/W) = Indicates the CMPB shadow FIFO is full. A CPU write will overwrite current shadow value.
8	SHDWAFULL	R	0h	Counter-compare A (CMPA) Shadow Register Full Status Flag. The flag bit is set when a 32 bit write to CMPA:CMPAHR register or a 16 bit write to CMPA register is made. A 16 bit write to CMPAHR register will not affect the flag. This bit self clears once a load-strobe occurs. 0h (R/W) = CMPA shadow FIFO not full yet 1h (R/W) = Indicates the CMPA shadow FIFO is full, a CPU write will overwrite the current shadow value.
7	RESERVED	R	0h	
6	SHDWBMODE	R/W	0h	Counter-compare B (CMPB) Register Operating Mode. 0h (R/W) = Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register. 1h (R/W) = Immediate mode. Only the active compare B register is used. All writes and reads directly access the active register for immediate compare action.
5	RESERVED	R	0h	
4	SHDWAMODE	R/W	0h	Counter-compare A (CMPA) Register Operating Mode. 0h (R/W) = Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register. 1h (R/W) = Immediate mode. Only the active compare register is used. All writes and reads directly access the active register for immediate compare action
3-2	LOADBMODE	R/W	0h	Active Counter-Compare B (CMPB) Load From Shadow Select Mode. This bit has no effect in immediate mode (CMPCTL[SHDWBMODE] = 1). 0h (R/W) = Load on CTR = 0 - Time-base counter equal to zero (TBCNT = 0000h) 1h (R/W) = Load on CTR = PRD - Time-base counter equal to period (TBCNT = TBPRD) 2h (R/W) = Load on either CTR = 0 or CTR = PRD 3h (R/W) = Freeze (no loads possible)

Table 15-70. CMPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	LOADAMODE	R/W	0h	Active Counter-Compare A (CMPA) Load From Shadow Select Mode. This bit has no effect in immediate mode (CMPCTL[SHDWAMODE] = 1). 0h (R/W) = Load on CTR = 0 - Time-base counter equal to zero (TBCNT = 0000h) 1h (R/W) = Load on CTR = PRD - Time-base counter equal to period (TBCNT = TBPRD) 2h (R/W) = Load on either CTR = 0 or CTR = PRD 3h (R/W) = Freeze (no loads possible)

15.2.4.8 CMPAHR Register (offset = 10h) [reset = 100h]

CMPAHR is shown in [Figure 15-78](#) and described in [Table 15-71](#).

This register is only available on ePWM instances that include the high-resolution PWM (HRPWM) extension; otherwise, this location is reserved.

Figure 15-78. CMPAHR Register

15	14	13	12	11	10	9	8
CMPAHR							
R/W-1h							
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-71. CMPAHR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	CMPAHR	R/W	1h	Compare A High-Resolution register bits for MEP step control. A minimum value of 1h is needed to enable HRPWM capabilities. Valid MEP range of operation 1-255h.
7-0	RESERVED	R	0h	

15.2.4.9 CMPA Register (offset = 12h) [reset = 0h]

 CMPA is shown in [Figure 15-79](#) and described in [Table 15-72](#).

Figure 15-79. CMPA Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPA															
R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-72. CMPA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	CMPA	R/W	0h	<p>The value in the active CMPA register is continuously compared to the time-base counter (TBCNT). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare A" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include the following.</p> <ul style="list-style-type: none"> (a) Do nothing the event is ignored. (b) Clear - Pull the EPWMxA and/or EPWMxB signal low. (c) Set - Pull the EPWMxA and/or EPWMxB signal high. (d) Toggle the EPWMxA and/or EPWMxB signal. <p>Shadowing of this register is enabled and disabled by the CMPCTL[SHDWAMODE] bit. By default this register is shadowed.</p> <ul style="list-style-type: none"> (a) If CMPCTL[SHDWAMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADAMODE] bit field determines which event will load the active register from the shadow register. (b) Before a write, the CMPCTL[SHDWAFULL] bit can be read to determine if the shadow register is currently full. (c) If CMPCTL[SHDWAMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. (d) In either mode, the active and shadow registers share the same memory map address.

15.2.4.10 CMPB Register (offset = 14h) [reset = 0h]

 CMPB is shown in [Figure 15-80](#) and described in [Table 15-73](#).

Figure 15-80. CMPB Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPB															
R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-73. CMPB Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	CMPB	R/W	0h	The value in the active CMPB register is continuously compared to the time-base counter (TBCNT). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare B" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include the following. (a) Do nothing, the event is ignored. (b) Clear - Pull the EPWMxA and/or EPWMxB signal low. (c) Set - Pull the EPWMxA and/or EPWMxB signal high. (d) Toggle the EPWMxA and/or EPWMxB signal. Shadowing of this register is enabled and disabled by the CMPCTL[SHDWBMODE] bit. By default this register is shadowed. (a) If CMPCTL[SHDWBMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADBMODE] bit field determines which event will load the active register from the shadow register: (b) Before a write, the CMPCTL[SHDWBFULL] bit can be read to determine if the shadow register is currently full. (c) If CMPCTL[SHDWBMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. (d) In either mode, the active and shadow registers share the same memory map address.

15.2.4.11 AQCTLA Register (offset = 16h) [reset = 0h]

 AQCTLA is shown in [Figure 15-81](#) and described in [Table 15-74](#).

Figure 15-81. AQCTLA Register

15	14	13	12	11	10	9	8
RESERVED				CBD		CBU	
R-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
CAD		CAU		PRD		ZRO	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-74. AQCTLA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	
11-10	CBD	R/W	0h	Action when the time-base counter equals the active CMPB register and the counter is decrementing. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxA output low. 2h (R/W) = Set - force EPWMxA output high. 3h (R/W) = Toggle EPWMxA output - low output signal will be forced high, and a high signal will be forced low.
9-8	CBU	R/W	0h	Action when the counter equals the active CMPB register and the counter is incrementing. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxA output low. 2h (R/W) = Set - force EPWMxA output high. 3h (R/W) = Toggle EPWMxA output - low output signal will be forced high, and a high signal will be forced low.
7-6	CAD	R/W	0h	Action when the counter equals the active CMPA register and the counter is decrementing. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxA output low. 2h (R/W) = Set - force EPWMxA output high. 3h (R/W) = Toggle EPWMxA output - low output signal will be forced high, and a high signal will be forced low.
5-4	CAU	R/W	0h	Action when the counter equals the active CMPA register and the counter is incrementing. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxA output low. 2h (R/W) = Set - force EPWMxA output high. 3h (R/W) = Toggle EPWMxA output - low output signal will be forced high, and a high signal will be forced low.
3-2	PRD	R/W	0h	Action when the counter equals the period. Note: By definition, in count up-down mode when the counter equals period the direction is defined as 0 or counting down. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxA output low. 2h (R/W) = Set - force EPWMxA output high. 3h (R/W) = Toggle EPWMxA output - low output signal will be forced high, and a high signal will be forced low.

Table 15-74. AQCTLA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	ZRO	R/W	0h	Action when counter equals zero. Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxA output low. 2h (R/W) = Set - force EPWMxA output high. 3h (R/W) = Toggle EPWMxA output - low output signal will be forced high, and a high signal will be forced low.

15.2.4.12 AQCTLB Register (offset = 18h) [reset = 0h]

 AQCTLB is shown in [Figure 15-82](#) and described in [Table 15-75](#).

Figure 15-82. AQCTLB Register

15	14	13	12	11	10	9	8
RESERVED				CBD		CBU	
R-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
CAD		CAU		PRD		ZRO	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-75. AQCTLB Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	
11-10	CBD	R/W	0h	Action when the counter equals the active CMPB register and the counter is decrementing. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxB output low. 2h (R/W) = Set - force EPWMxB output high. 3h (R/W) = Toggle EPWMxB output - low output signal will be forced high, and a high signal will be forced low.
9-8	CBU	R/W	0h	Action when the counter equals the active CMPB register and the counter is incrementing. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxB output low. 2h (R/W) = Set - force EPWMxB output high. 3h (R/W) = Toggle EPWMxB output - low output signal will be forced high, and a high signal will be forced low.
7-6	CAD	R/W	0h	Action when the counter equals the active CMPA register and the counter is decrementing. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxB output low. 2h (R/W) = Set - force EPWMxB output high. 3h (R/W) = Toggle EPWMxB output - low output signal will be forced high, and a high signal will be forced low.
5-4	CAU	R/W	0h	Action when the counter equals the active CMPA register and the counter is incrementing. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxB output low. 2h (R/W) = Set - force EPWMxB output high. 3h (R/W) = Toggle EPWMxB output - low output signal will be forced high, and a high signal will be forced low.
3-2	PRD	R/W	0h	Action when the counter equals the period. Note: By definition, in count up-down mode when the counter equals period the direction is defined as 0 or counting down. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxB output low. 2h (R/W) = Set - force EPWMxB output high. 3h (R/W) = Toggle EPWMxB output - low output signal will be forced high, and a high signal will be forced low.

Table 15-75. AQCTLB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	ZRO	R/W	0h	Action when counter equals zero. Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Clear - force EPWMxB output low. 2h (R/W) = Set - force EPWMxB output high. 3h (R/W) = Toggle EPWMxB output - low output signal will be forced high, and a high signal will be forced low.

15.2.4.13 AQSFR Register (offset = 1Ah) [reset = 0h]

 AQSFR is shown in [Figure 15-83](#) and described in [Table 15-76](#).

Figure 15-83. AQSFR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RLDCSF		OTSFB	ACTSFB		OTSFA	ACTSFA	
R/W-0h		R/W-0h	R/W-0h		R/W-0h	R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-76. AQSFR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	
7-6	RLDCSF	R/W	0h	AQCSFRC Active Register Reload From Shadow Options. 0h (R/W) = Load on event counter equals zero 1h (R/W) = Load on event counter equals period 2h (R/W) = Load on event counter equals zero or counter equals period 3h (R/W) = Load immediately (the active register is directly accessed by the CPU and is not loaded from the shadow register).
5	OTSFB	R/W	0h	One-Time Software Forced Event on Output B. 0h (R/W) = Writing a 0 (zero) has no effect. Always reads back a 0. This bit is auto cleared once a write to this register is complete, that is, a forced event is initiated. This is a one-shot forced event. It can be overridden by another subsequent event on output B. 1h (R/W) = Initiates a single s/w forced event
4-3	ACTSFB	R/W	0h	Action when One-Time Software Force B Is invoked 0h (R/W) = Does nothing (action disabled) 1h (R/W) = Clear (low) 2h (R/W) = Set (high) 3h (R/W) = Toggle (Low -> High, High -> Low). Note: This action is not qualified by counter direction (CNT_dir)
2	OTSFA	R/W	0h	One-Time Software Forced Event on Output A. 0h (R/W) = Writing a 0 (zero) has no effect. Always reads back a 0. This bit is auto cleared once a write to this register is complete (that is, a forced event is initiated). 1h (R/W) = Initiates a single software forced event.
1-0	ACTSFA	R/W	0h	Action When One-Time Software Force A Is Invoked. 0h (R/W) = Does nothing (action disabled). 1h (R/W) = Clear (low). 2h (R/W) = Set (high). 3h (R/W) = Toggle (Low -> High, High -> Low). Note: This action is not qualified by counter direction (CNT_dir)

15.2.4.14 AQCSFRC Register (offset = 1Ch) [reset = 0h]

 AQCSFRC is shown in [Figure 15-84](#) and described in [Table 15-77](#).

Figure 15-84. AQCSFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CSFB		CSFA	
R-0h				R/W-0h		R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-77. AQCSFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	
3-2	CSFB	R/W	0h	Continuous Software Force on Output B. In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register. To configure shadow mode, use AQSFRC[RLDCSF]. 0h (R/W) = Forcing disabled, that is, has no effect 1h (R/W) = Forces a continuous low on output B 2h (R/W) = Forces a continuous high on output B 3h (R/W) = Software forcing is disabled and has no effect
1-0	CSFA	R/W	0h	Continuous Software Force on Output A In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register. 0h (R/W) = Forcing disabled, that is, has no effect 1h (R/W) = Forces a continuous low on output A 2h (R/W) = Forces a continuous high on output A 3h (R/W) = Software forcing is disabled and has no effect

15.2.4.15 DBCTL Register (offset = 1Eh) [reset = 0h]

 DBCTL is shown in [Figure 15-85](#) and described in [Table 15-78](#).

Figure 15-85. DBCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		IN_MODE		POLSEL		OUT_MODE	
R-0h		R/W-0h		R/W-0h		R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-78. DBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R	0h	
5-4	IN_MODE	R/W	0h	Dead Band Input Mode Control. Bit 5 controls the S5 switch and bit 4 controls the S4 switch. This allows you to select the input source to the falling-edge and rising-edge delay. To produce classical dead-band waveforms, the default is EPWMxA In is the source for both falling and rising-edge delays. 0h (R/W) = EPWMxA In (from the action-qualifier) is the source for both falling-edge and rising-edge delay. 1h (R/W) = EPWMxB In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxA In (from the action-qualifier) is the source for falling-edge delayed signal. 2h (R/W) = EPWMxA In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxB In (from the action-qualifier) is the source for falling-edge delayed signal. 3h (R/W) = EPWMxB In (from the action-qualifier) is the source for both rising-edge delay and falling-edge delayed signal.
3-2	POLSEL	R/W	0h	Polarity Select Control. Bit 3 controls the S3 switch and bit 2 controls the S2 switch. This allows you to selectively invert one of the delayed signals before it is sent out of the dead-band submodule. The following descriptions correspond to classical upper/lower switch control as found in one leg of a digital motor control inverter. These assume that DBCTL[OUT_MODE] = 1,1 and DBCTL[IN_MODE] = 0,0. Other enhanced modes are also possible, but not regarded as typical usage modes. 0h (R/W) = Active high (AH) mode. Neither EPWMxA nor EPWMxB is inverted (default). 1h (R/W) = Active low complementary (ALC) mode. EPWMxA is inverted. 2h (R/W) = Active high complementary (AHC). EPWMxB is inverted. 3h (R/W) = Active low (AL) mode. Both EPWMxA and EPWMxB are inverted.

Table 15-78. DBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	OUT_MODE	R/W	0h	<p>Dead-band Output Mode Control. Bit 1 controls the S1 switch and bit 0 controls the S0 switch. This allows you to selectively enable or bypass the dead-band generation for the falling-edge and rising-edge delay.</p> <p>0h (R/W) = Dead-band generation is bypassed for both output signals. In this mode, both the EPWMxA and EPWMxB output signals from the action-qualifier are passed directly to the PWM-chopper submodule. In this mode, the POLSEL and IN_MODE bits have no effect.</p> <p>1h (R/W) = Disable rising-edge delay. The EPWMxA signal from the action-qualifier is passed straight through to the EPWMxA input of the PWM-chopper submodule. The falling-edge delayed signal is seen on output EPWMxB. The input signal for the delay is determined by DBCTL[IN_MODE].</p> <p>2h (R/W) = Disable falling-edge delay. The EPWMxB signal from the action-qualifier is passed straight through to the EPWMxB input of the PWM-chopper submodule. The rising-edge delayed signal is seen on output EPWMxA. The input signal for the delay is determined by DBCTL[IN_MODE].</p> <p>3h (R/W) = Dead-band is fully enabled for both rising-edge delay on output EPWMxA and falling-edge delay on output EPWMxB. The input signal for the delay is determined by DBCTL[IN_MODE].</p>

15.2.4.16 DBRED Register (offset = 20h) [reset = 0h]

DBRED is shown in [Figure 15-86](#) and described in [Table 15-79](#).

Figure 15-86. DBRED Register

15	14	13	12	11	10	9	8
RESERVED						DEL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
DEL							
R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-79. DBRED Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	
9-0	DEL	R/W	0h	Rising Edge Delay Count. 10 bit counter.

15.2.4.17 DBFED Register (offset = 22h) [reset = 0h]

DBFED is shown in [Figure 15-87](#) and described in [Table 15-80](#).

Figure 15-87. DBFED Register

15	14	13	12	11	10	9	8
RESERVED						DEL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
DEL							
R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-80. DBFED Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	
9-0	DEL	R/W	0h	Falling Edge Delay Count. 10 bit counter

15.2.4.18 TZSEL Register (offset = 24h) [reset = 0h]

 TZSEL is shown in [Figure 15-88](#) and described in [Table 15-81](#).

Figure 15-88. TZSEL Register

15	14	13	12	11	10	9	8
OSHTn							
R/W-0h							
7	6	5	4	3	2	1	0
CBCn							
R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-81. TZSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	OSHTn	R/W	0h	Trip-zone n (TZn) select. One-Shot (OSHT) trip-zone enable/disable. When any of the enabled pins go low, a one-shot trip event occurs for this ePWM module. When the event occurs, the action defined in the TZCTL register is taken on the EPWMxA and EPWMxB outputs. The one-shot trip condition remains latched until you clear the condition via the TZCLR register. 0h (R/W) = Disable TZn as a one-shot trip source for this ePWM module. 1h (R/W) = Enable TZn as a one-shot trip source for this ePWM module.
7-0	CBCn	R/W	0h	Trip-zone n (TZn) select. Cycle-by-Cycle (CBC) trip-zone enable/disable. When any of the enabled pins go low, a cycle-by-cycle trip event occurs for this ePWM module. When the event occurs, the action defined in the TZCTL register is taken on the EPWMxA and EPWMxB outputs. A cycle-by-cycle trip condition is automatically cleared when the time-base counter reaches zero. 0h (R/W) = Disable TZn as a CBC trip source for this ePWM module. 1h (R/W) = Enable TZn as a CBC trip source for this ePWM module.

15.2.4.19 TZCTL Register (offset = 28h) [reset = 0h]

 TZCTL is shown in [Figure 15-89](#) and described in [Table 15-82](#).

Figure 15-89. TZCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TZB		TZA	
R-0h				R/W-0h		R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-82. TZCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	
3-2	TZB	R/W	0h	When a trip event occurs the following action is taken on output EPWMxB. Which trip-zone pins can cause an event is defined in the TZSEL register. 0h (R/W) = High impedance (EPWMxB = High-impedance state) 1h (R/W) = Force EPWMxB to a high state 2h (R/W) = Force EPWMxB to a low state 3h (R/W) = Do nothing, no action is taken on EPWMxB.
1-0	TZA	R/W	0h	When a trip event occurs the following action is taken on output EPWMxA. Which trip-zone pins can cause an event is defined in the TZSEL register. 0h (R/W) = High impedance (EPWMxA = High-impedance state) 1h (R/W) = Force EPWMxA to a high state 2h (R/W) = Force EPWMxA to a low state 3h (R/W) = Do nothing, no action is taken on EPWMxA.

15.2.4.20 TZEINT Register (offset = 2Ah) [reset = 0h]

TZEINT is shown in [Figure 15-90](#) and described in [Table 15-83](#).

Figure 15-90. TZEINT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					OST	CBC	RESERVED
R-0h					R/W-0h	R/W-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-83. TZEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	
2	OST	R/W	0h	Trip-zone One-Shot Interrupt Enable 0h (R/W) = Disable one-shot interrupt generation 1h (R/W) = Enable Interrupt generation; a one-shot trip event will cause a EPWMxTZINT interrupt.
1	CBC	R/W	0h	Trip-zone Cycle-by-Cycle Interrupt Enable 0h (R/W) = Disable cycle-by-cycle interrupt generation. 1h (R/W) = Enable interrupt generation; a cycle-by-cycle trip event will cause an EPWMxTZINT interrupt.
0	RESERVED	R	0h	

15.2.4.21 TZFLG Register (offset = 2Ch) [reset = 0h]

TZFLG is shown in [Figure 15-91](#) and described in [Table 15-84](#).

Figure 15-91. TZFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					OST	CBC	INT
R-0h					R-0h	R-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-84. TZFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	
2	OST	R	0h	Latched Status Flag for A One-Shot Trip Event. 0h (R/W) = No one-shot trip event has occurred. 1h (R/W) = Indicates a trip event has occurred on a pin selected as a one-shot trip source. This bit is cleared by writing the appropriate value to the TZCLR register.
1	CBC	R	0h	Latched Status Flag for Cycle-By-Cycle Trip Event 0h (R/W) = No cycle-by-cycle trip event has occurred. 1h (R/W) = Indicates a trip event has occurred on a pin selected as a cycle-by-cycle trip source. The TZFLG[CBC] bit will remain set until it is manually cleared by the user. If the cycle-by-cycle trip event is still present when the CBC bit is cleared, then CBC will be immediately set again. The specified condition on the pins is automatically cleared when the ePWM time-base counter reaches zero (TBCNT = 0000h) if the trip condition is no longer present. The condition on the pins is only cleared when the TBCNT = 0000h no matter where in the cycle the CBC flag is cleared. This bit is cleared by writing the appropriate value to the TZCLR register.
0	INT	R	0h	Latched Trip Interrupt Status Flag 0h (R/W) = Indicates no interrupt has been generated. 1h (R/W) = Indicates an EPWMxTZINT interrupt was generated because of a trip condition. No further EPWMxTZINT interrupts will be generated until this flag is cleared. If the interrupt flag is cleared when either CBC or OST is set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts. This bit is cleared by writing the appropriate value to the TZCLR register.

15.2.4.22 TZCLR Register (offset = 2Eh) [reset = 0h]

 TZCLR is shown in [Figure 15-92](#) and described in [Table 15-85](#).

Figure 15-92. TZCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					OST	CBC	INT
R-0h					R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-85. TZCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	
2	OST	R/W	0h	Clear Flag for One-Shot Trip (OST) Latch 0h (R/W) = Has no effect. Always reads back a 0. 1h (R/W) = Clears this Trip (set) condition.
1	CBC	R/W	0h	Clear Flag for Cycle-By-Cycle (CBC) Trip Latch 0h (R/W) = Has no effect. Always reads back a 0. 1h (R/W) = Clears this Trip (set) condition.
0	INT	R/W	0h	Global Interrupt Clear Flag 0h (R/W) = Has no effect. Always reads back a 0. 1h (R/W) = Clears the trip-interrupt flag for this ePWM module (TZFLG[INT]). Note: No further EPWMxTZINT interrupts will be generated until the flag is cleared. If the TZFLG[INT] bit is cleared and any of the other flag bits are set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts.

15.2.4.23 TZFRC Register (offset = 30h) [reset = 0h]

 TZFRC is shown in [Figure 15-93](#) and described in [Table 15-86](#).

Figure 15-93. TZFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					OST	CBC	RESERVED
R-0h					R/W-0h	R/W-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-86. TZFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	
2	OST	R/W	0h	Force a One-Shot Trip Event via Software 0h (R/W) = Writing of 0 is ignored. Always reads back a 0. 1h (R/W) = Forces a one-shot trip event and sets the TZFLG[OST] bit.
1	CBC	R/W	0h	Force a Cycle-by-Cycle Trip Event via Software 0h (R/W) = Writing of 0 is ignored. Always reads back a 0. 1h (R/W) = Forces a cycle-by-cycle trip event and sets the TZFLG[CBC] bit.
0	RESERVED	R	0h	

15.2.4.24 ETSEL Register (offset = 32h) [reset = 0h]

 ETSEL is shown in [Figure 15-94](#) and described in [Table 15-87](#).

Figure 15-94. ETSEL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				INTEN	INTSEL		
R-0h				R/W-0h	R/W-0h		

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-87. ETSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	
3	INTEN	R/W	0h	Enable ePWM Interrupt (EPWMx_INT) Generation 0h (R/W) = Disable EPWMx_INT generation 1h (R/W) = Enable EPWMx_INT generation
2-0	INTSEL	R/W	0h	ePWM Interrupt (EPWMx_INT) Selection Options 0h (R/W) = Reserved 1h (R/W) = Enable event time-base counter equal to zero. (TBCNT = 0000h) 2h (R/W) = Enable event time-base counter equal to period (TBCNT = TBPRD) 3h (R/W) = Reserved 4h (R/W) = Enable event time-base counter equal to CMPA when the timer is incrementing. 5h (R/W) = Enable event time-base counter equal to CMPA when the timer is decrementing. 6h (R/W) = Enable event - time-base counter equal to CMPB when the timer is incrementing. 7h (R/W) = Enable event - time-base counter equal to CMPB when the timer is decrementing.

15.2.4.25 ETPS Register (offset = 34h) [reset = 0h]

ETPS is shown in [Figure 15-95](#) and described in [Table 15-88](#).

Figure 15-95. ETPS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				INTCNT		INTPRD	
R-0h				R-0h		R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-88. ETPS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	
3-2	INTCNT	R	0h	<p>ePWM Interrupt Event (EPWMx_INT) Counter Register. These bits indicate how many selected ETSEL[INTSEL] events have occurred. These bits are automatically cleared when an interrupt pulse is generated.</p> <p>If interrupts are disabled, ETSEL[INT] = 0 or the interrupt flag is set, ETFLG[INT] = 1, the counter will stop counting events when it reaches the period value ETPS[INTCNT] = ETPS[INTPRD].</p> <p>0h (R/W) = No events have occurred. 1h (R/W) = 1 event has occurred. 2h (R/W) = 2 events have occurred. 3h (R/W) = 3 events have occurred.</p>
1-0	INTPRD	R/W	0h	<p>ePWM Interrupt (EPWMx_INT) Period Select. These bits determine how many selected ETSEL[INTSEL] events need to occur before an interrupt is generated. To be generated, the interrupt must be enabled (ETSEL[INT] = 1). If the interrupt status flag is set from a previous interrupt (ETFLG[INT] = 1) then no interrupt will be generated until the flag is cleared via the ETCLR[INT] bit. This allows for one interrupt to be pending while another is still being serviced. Once the interrupt is generated, the ETPS[INTCNT] bits will automatically be cleared. Writing a INTPRD value that is the same as the current counter value will trigger an interrupt if it is enabled and the status flag is clear. Writing a INTPRD value that is less than the current counter value will result in an undefined state. If a counter event occurs at the same instant as a new zero or non-zero INTPRD value is written, the counter is incremented.</p> <p>0h (R/W) = Disable the interrupt event counter. No interrupt will be generated and ETFRC[INT] is ignored. 1h (R/W) = Generate an interrupt on the first event INTCNT = 01 (first event) 2h (R/W) = Generate interrupt on ETPS[INTCNT] = 1,0 (second event) 3h (R/W) = Generate interrupt on ETPS[INTCNT] = 1,1 (third event)</p>

15.2.4.26 ETFLG Register (offset = 36h) [reset = 0h]

 ETFLG is shown in [Figure 15-96](#) and described in [Table 15-89](#).

Figure 15-96. ETFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INT
R-0h							R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-89. ETFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	
0	INT	R	0h	Latched ePWM Interrupt (EPWMx_INT) Status Flag 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates that an ePWMx interrupt (EWPMx_INT) was generated. No further interrupts will be generated until the flag bit is cleared. Up to one interrupt can be pending while the ETFLG[INT] bit is still set. If an interrupt is pending, it will not be generated until after the ETFLG[INT] bit is cleared.

15.2.4.27 ETCLR Register (offset = 38h) [reset = 0h]

ETCLR is shown in [Figure 15-97](#) and described in [Table 15-90](#).

Figure 15-97. ETCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INT
R-0h							R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-90. ETCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	
0	INT	R	0h	ePWM Interrupt (EPWMx_INT) Flag Clear Bit 0h (R/W) = Writing a 0 has no effect. Always reads back a 0. 1h (R/W) = Clears the ETFLG[INT] flag bit and enable further interrupts pulses to be generated. NOTE: Interrupts can also used as DMA events, and this will also enable further DMA events to be generated

15.2.4.28 ETFRC Register (offset = 3Ah) [reset = 0h]

 ETFRC is shown in [Figure 15-98](#) and described in [Table 15-91](#).

Figure 15-98. ETFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INT
R-0h							R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-91. ETFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	
0	INT	R	0h	INT Force Bit. The interrupt will only be generated if the event is enabled in the ETSEL register. The INT flag bit will be set regardless. 0h (R/W) = Writing 0 to this bit will be ignored. Always reads back a 0. 1h (R/W) = Generates an interrupt on EPWMxINT and set the INT flag bit. This bit is used for test purposes.

15.2.4.29 PCCTL Register (offset = 3Ch) [reset = 0h]

PCCTL is shown in [Figure 15-99](#) and described in [Table 15-92](#).

Figure 15-99. PCCTL Register

15	14	13	12	11	10	9	8
RESERVED					CHPDUTY		
R-0h					R/W-0h		
7	6	5	4	3	2	1	0
CHPFREQ			OSHTWTH			CHPEN	
R/W-0h			R/W-0h			R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-92. PCCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	
10-8	CHPDUTY	R/W	0h	Chopping Clock Duty Cycle 0h (R/W) = Duty = 1/8 (12.5%) 1h (R/W) = Duty = 2/8 (25.0%) 2h (R/W) = Duty = 3/8 (37.5%) 3h (R/W) = Duty = 4/8 (50.0%) 4h (R/W) = Duty = 5/8 (62.5%) 5h (R/W) = Duty = 6/8 (75.0%) 6h (R/W) = Duty = 7/8 (87.5%) 7h (R/W) = Reserved.
7-5	CHPFREQ	R/W	0h	Chopping Clock Frequency 0h (R/W) = Divide by 1 (no prescale). 1h (R/W) = Divide by 2. 2h (R/W) = Divide by 3. 3h (R/W) = Divide by 4. 4h (R/W) = Divide by 5. 5h (R/W) = Divide by 6. 6h (R/W) = Divide by 7. 7h (R/W) = Divide by 8.
4-1	OSHTWTH	R/W	0h	One-Shot Pulse Width 0h (R/W) = 1 - SYSCLKOUT/8 wide 1h (R/W) = 2 - SYSCLKOUT/8 wide 2h (R/W) = 3 - SYSCLKOUT/8 wide 3h (R/W) = 4 - SYSCLKOUT/8 wide Fh (R/W) = 16 - SYSCLKOUT/8 wide
0	CHPEN	R/W	0h	PWM-chopping Enable 0h (R/W) = Disable (bypass) PWM chopping function 1h (R/W) = Enable chopping function

15.2.4.30 HRCNFG Register (offset = C0h) [reset = 0h]

HRCNFG is shown in [Figure 15-100](#) and described in [Table 15-93](#).

This register is only available on ePWM instances that include the high-resolution PWM (HRPWM) extension; otherwise, this location is reserved.

Figure 15-100. HRCNFG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				HRLOAD	CTLMODE	EDGMODE	
R-0h				R/W-0h	R/W-0h	R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-93. HRCNFG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	
3	HRLOAD	R/W	0h	Shadow mode bit - Selects the time event that loads the CMPAHR shadow value into the active register. Note: Load mode selection is valid only if CTLMODE = 0 has been selected. You should select this event to match the selection of the CMPA load mode (CMPCTL[LOADMODE] bits) in the EPWM module as follows: 0x 0: Load on CTR = 0 Time-base counter equal to zero (TBCNT = 0000h) 0x 1: Load on CTR = PRD Time-base counter equal to period (TBCNT = TBPRD) 0x 2: Load on either CTR = 0 or CTR = PRD (should not be used with HRPWM) 0x 3: Freeze (no loads possible should not be used with HRPWM) 0h (R/W) = CTR = PRD (counter equal period) 1h (R/W) = CTR = 0 (counter equals zero)
2	CTLMODE	R/W	0h	Control Mode Bits - Selects the register (CMP or TBPHS) that controls the MEP. 0h (R/W) = CMPAHR(8) Register controls the edge position (this is duty control mode). (default on reset) 1h (R/W) = TBPHSHR(8) Register controls the edge position (this is phase control mode).
1-0	EDGMODE	R/W	0h	Edge Mode Bits - Selects the edge of the PWM that is controlled by the micro-edge position (MEP) logic. 0h (R/W) = HRPWM capability is disabled (default on reset) 1h (R/W) = MEP control of rising edge 2h (R/W) = MEP control of falling edge 3h (R/W) = MEP control of both edges

15.3 Enhanced Capture (eCAP) Module

15.3.1 Introduction

15.3.1.1 Purpose of the Peripheral

Uses for eCAP include:

- Sample rate measurements of audio inputs
- Speed measurements of rotating machinery (for example, toothed sprockets sensed via Hall sensors)
- Elapsed time measurements between position sensor pulses
- Period and duty cycle measurements of pulse train signals
- Decoding current or voltage amplitude derived from duty cycle encoded current/voltage sensors

15.3.1.2 Features

The eCAP module includes the following features:

- 32-bit time base counter
- 4-event time-stamp registers (each 32 bits)
- Edge polarity selection for up to four sequenced time-stamp capture events
- Interrupt on either of the four events
- Single shot capture of up to four event time-stamps
- Continuous mode capture of time-stamps in a four-deep circular buffer
- Absolute time-stamp capture
- Difference (Delta) mode time-stamp capture
- All above resources dedicated to a single input pin
- When not used in capture mode, the ECAP module can be configured as a single channel PWM output

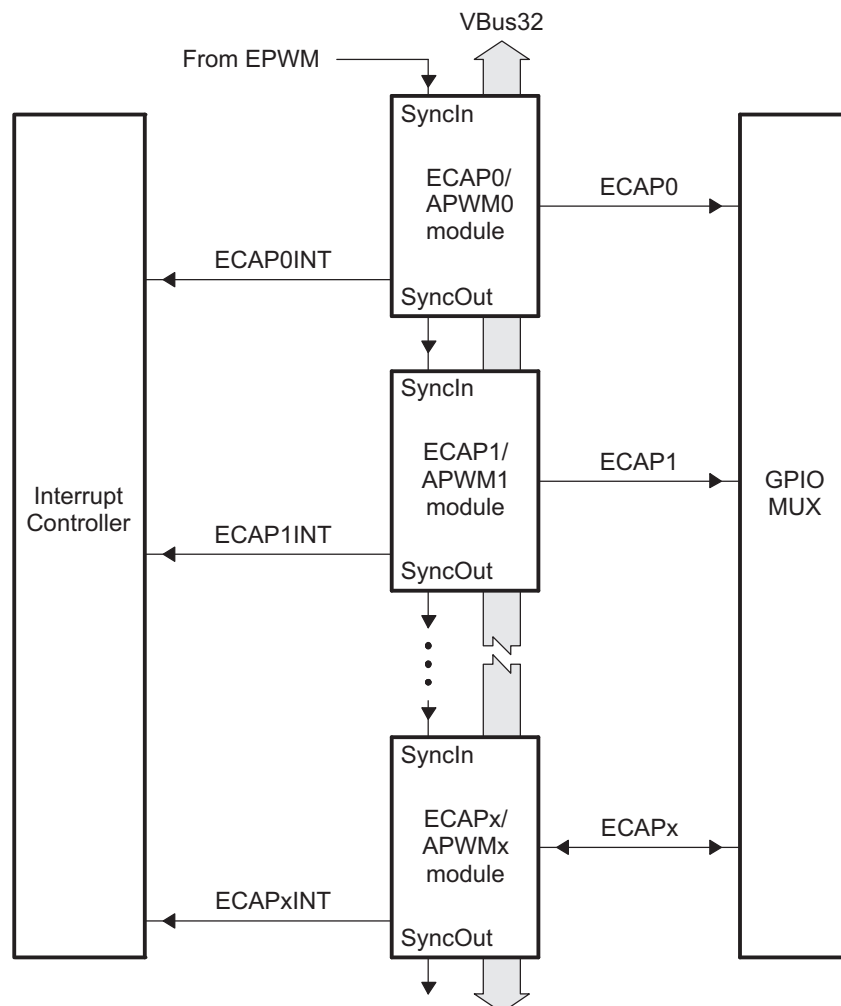
15.3.2 Functional Description

The eCAP module represents one complete capture channel that can be instantiated multiple times depending on the target device. In the context of this guide, one eCAP channel has the following independent key resources:

- Dedicated input capture pin
- 32-bit time base counter
- 4 × 32-bit time-stamp capture registers (CAP1-CAP4)
- 4-stage sequencer (Modulo4 counter) that is synchronized to external events, ECAP pin rising/falling edges.
- Independent edge polarity (rising/falling edge) selection for all 4 events
- Input capture signal prescaling (from 2-62)
- One-shot compare register (2 bits) to freeze captures after 1 to 4 time-stamp events
- Control for continuous time-stamp captures using a 4-deep circular buffer (CAP1-CAP4) scheme
- Interrupt capabilities on any of the 4 capture events

Multiple identical eCAP modules can be contained in a system as shown in [Figure 15-101](#). The number of modules is device-dependent and is based on target application needs. In this chapter, the letter x within a signal or module name is used to indicate a generic eCAP instance on a device.

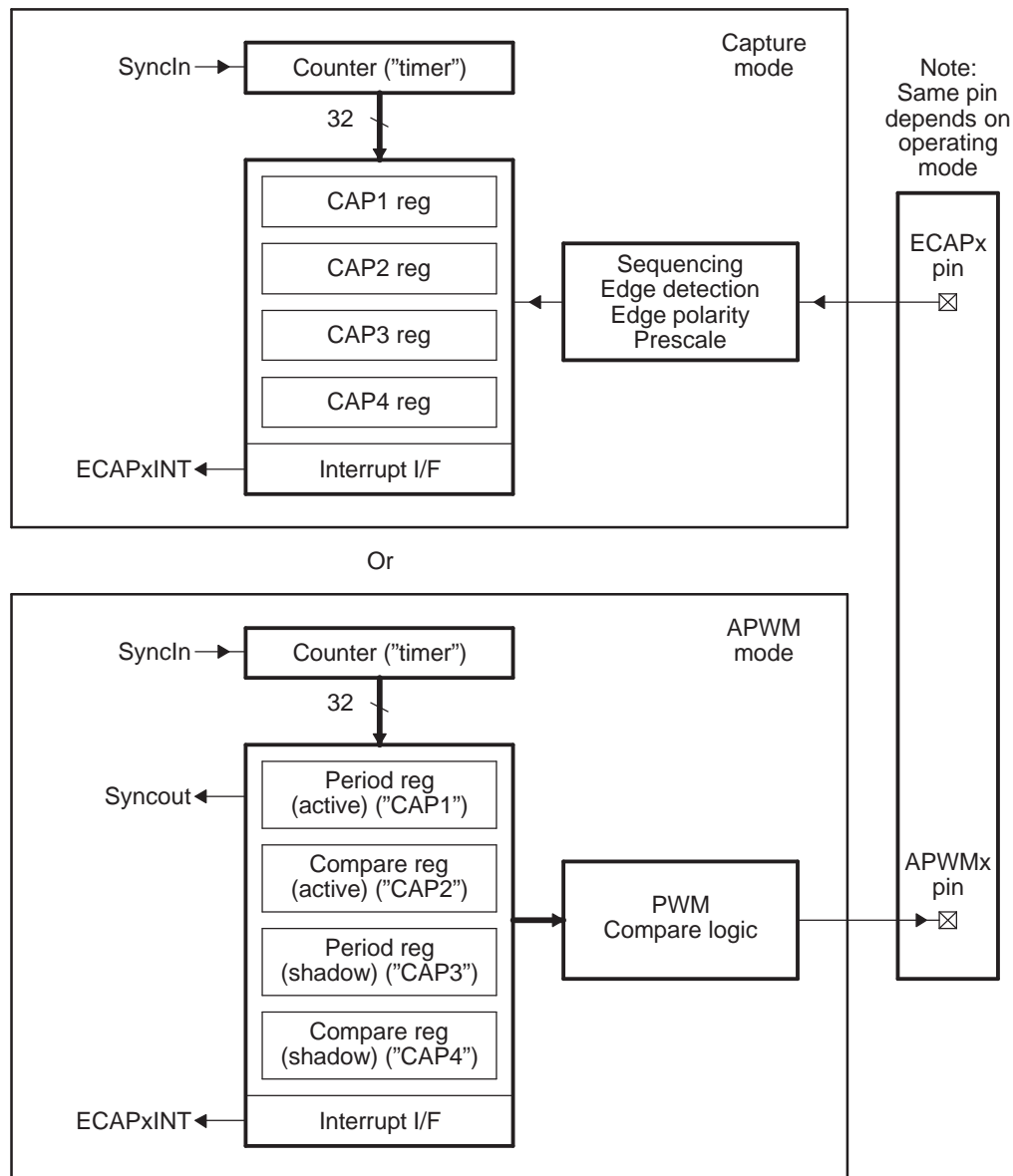
Figure 15-101. Multiple eCAP Modules



15.3.2.1 Capture and APWM Operating Mode

You can use the eCAP module resources to implement a single-channel PWM generator (with 32 bit capabilities) when it is not being used for input captures. The counter operates in count-up mode, providing a time-base for asymmetrical pulse width modulation (PWM) waveforms. The CAP1 and CAP2 registers become the active period and compare registers, respectively, while CAP3 and CAP4 registers become the period and capture shadow registers, respectively. Figure 15-102 is a high-level view of both the capture and auxiliary pulse-width modulator (APWM) modes of operation.

Figure 15-102. Capture and APWM Modes of Operation

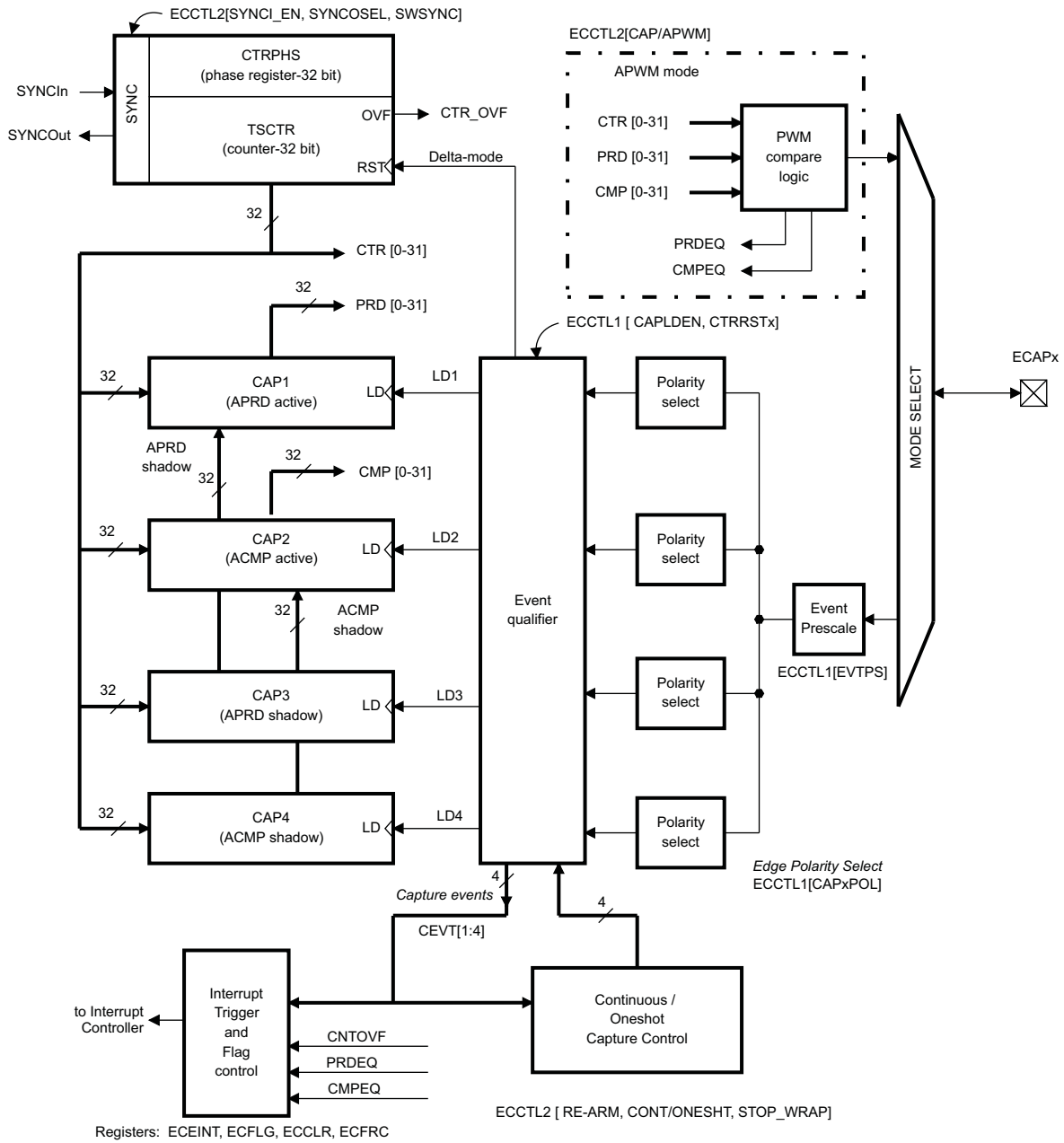


- (1) A single pin is shared between CAP and APWM functions. In capture mode, it is an input; in APWM mode, it is an output.
- (2) In APWM mode, writing any value to CAP1/CAP2 active registers also writes the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 invokes the shadow mode.

15.3.2.2 Capture Mode Description

Figure 15-103 shows the various components that implement the capture function.

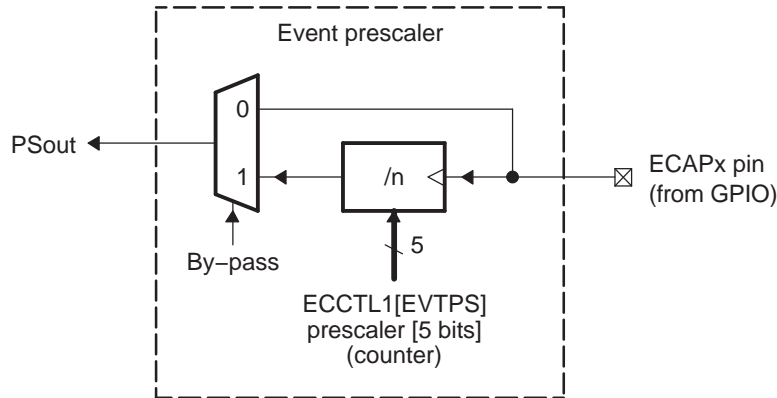
Figure 15-103. Capture Function Diagram



15.3.2.2.1 Event Prescaler

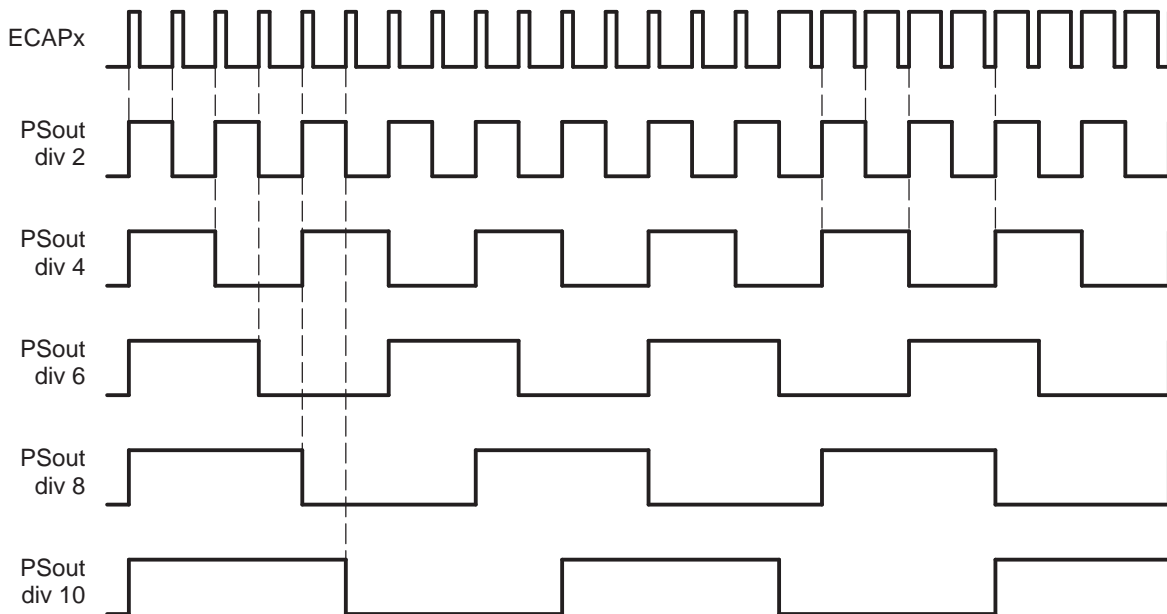
An input capture signal (pulse train) can be prescaled by $N = 2-62$ (in multiples of 2) or can bypass the prescaler. This is useful when very high frequency signals are used as inputs. Figure 15-104 shows a functional diagram and Figure 15-105 shows the operation of the prescale function.

Figure 15-104. Event Prescale Control



- (1) When a prescale value of 1 is chosen ($ECCTL1[13:9] = 0000$) the input capture signal by-passes the prescale logic completely.

Figure 15-105. Prescale Function Waveforms



15.3.2.2.2 Edge Polarity Select and Qualifier

- Four independent edge polarity (rising edge/falling edge) selection multiplexers are used, one for each capture event.
- Each edge (up to 4) is event qualified by the Modulo4 sequencer.
- The edge event is gated to its respective CAP n register by the Mod4 counter. The CAP n register is loaded on the falling edge.

15.3.2.2.3 Continuous/One-Shot Control

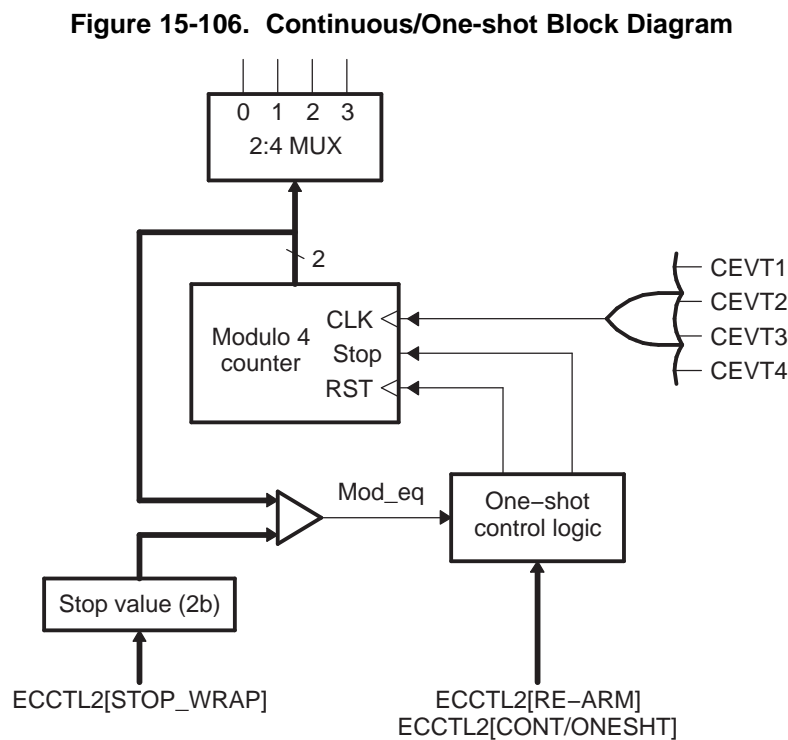
- The Mod4 (2 bit) counter is incremented via edge qualified events (CEVT1-CEVT4).
- The Mod4 counter continues counting (0->1->2->3->0) and wraps around unless stopped.
- A 2-bit stop register is used to compare the Mod4 counter output, and when equal stops the Mod4 counter and inhibits further loads of the CAP1-CAP4 registers. This occurs during one-shot operation.

The continuous/one-shot block (Figure 15-106) controls the start/stop and reset (zero) functions of the Mod4 counter via a mono-shot type of action that can be triggered by the stop-value comparator and re-armed via software control.

Once armed, the eCAP module waits for 1-4 (defined by stop-value) capture events before freezing both the Mod4 counter and contents of CAP1-4 registers (time-stamps).

Re-arming prepares the eCAP module for another capture sequence. Also re-arming clears (to zero) the Mod4 counter and permits loading of CAP1-4 registers again, providing the CAPLDEN bit is set.

In continuous mode, the Mod4 counter continues to run (0->1->2->3->0, the one-shot action is ignored, and capture values continue to be written to CAP1-4 in a circular buffer sequence.



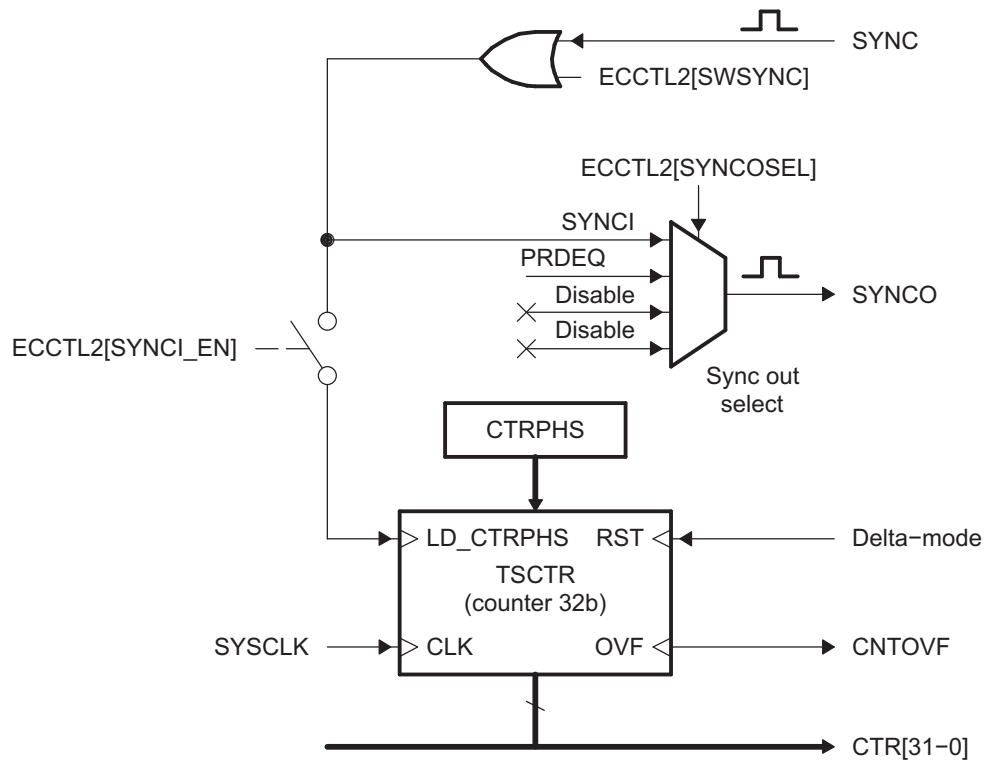
15.3.2.2.4 32-Bit Counter and Phase Control

This counter (Figure 15-107) provides the time-base for event captures, and is clocked via the system clock.

A phase register is provided to achieve synchronization with other counters, via a hardware and software forced sync. This is useful in APWM mode when a phase offset between modules is needed.

On any of the four event loads, an option to reset the 32-bit counter is given. This is useful for time difference capture. The 32-bit counter value is captured first, then it is reset to 0 by any of the LD1-LD4 signals.

Figure 15-107. Counter and Synchronization Block Diagram



15.3.2.2.5 CAP1-CAP4 Registers

These 32-bit registers are fed by the 32-bit counter timer bus, CTR[0-31] and are loaded (capture a time-stamp) when their respective LD inputs are strobed.

Loading of the capture registers can be inhibited via control bit CAPLDEN. During one-shot operation, this bit is cleared (loading is inhibited) automatically when a stop condition occurs, StopValue = Mod4.

CAP1 and CAP2 registers become the active period and compare registers, respectively, in APWM mode.

CAP3 and CAP4 registers become the respective shadow registers (APRD and ACMP) for CAP1 and CAP2 during APWM operation.

15.3.2.2.6 Interrupt Control

An Interrupt can be generated on capture events (CEVT1-CEVT4, CINTOVF) or APWM events (PRDEQ, CMPEQ). See [Figure 15-108](#).

A counter overflow event (FFFF FFFFh->0000 0000h) is also provided as an interrupt source (CINTOVF).

The capture events are edge and sequencer qualified (that is, ordered in time) by the polarity select and Mod4 gating, respectively.

One of these events can be selected as the interrupt source (from the eCAP n module) going to the interrupt controller.

Seven interrupt events (CEVT1, CEVT2, CEVT3, CEVT4, CINTOVF, PRDEQ, CMPEQ) can be generated. The interrupt enable register (ECEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (ECFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit (INT). An interrupt pulse is generated to the interrupt controller only if any of the interrupt events are enabled, the flag bit is 1, and the INT flag bit is 0. The interrupt service routine must clear the global interrupt flag bit and the serviced event via the interrupt clear register (ECCLR) before any other interrupt pulses are generated. You can force an interrupt event via the interrupt force register (ECFRC). This is useful for test purposes.

Note that the interrupts coming from the eCAP module are also used as DMA events. The interrupt registers should be used to enable and clear the current DMA event in order for the eCAP module to generate subsequent DMA events.

15.3.2.2.7 Shadow Load and Lockout Control

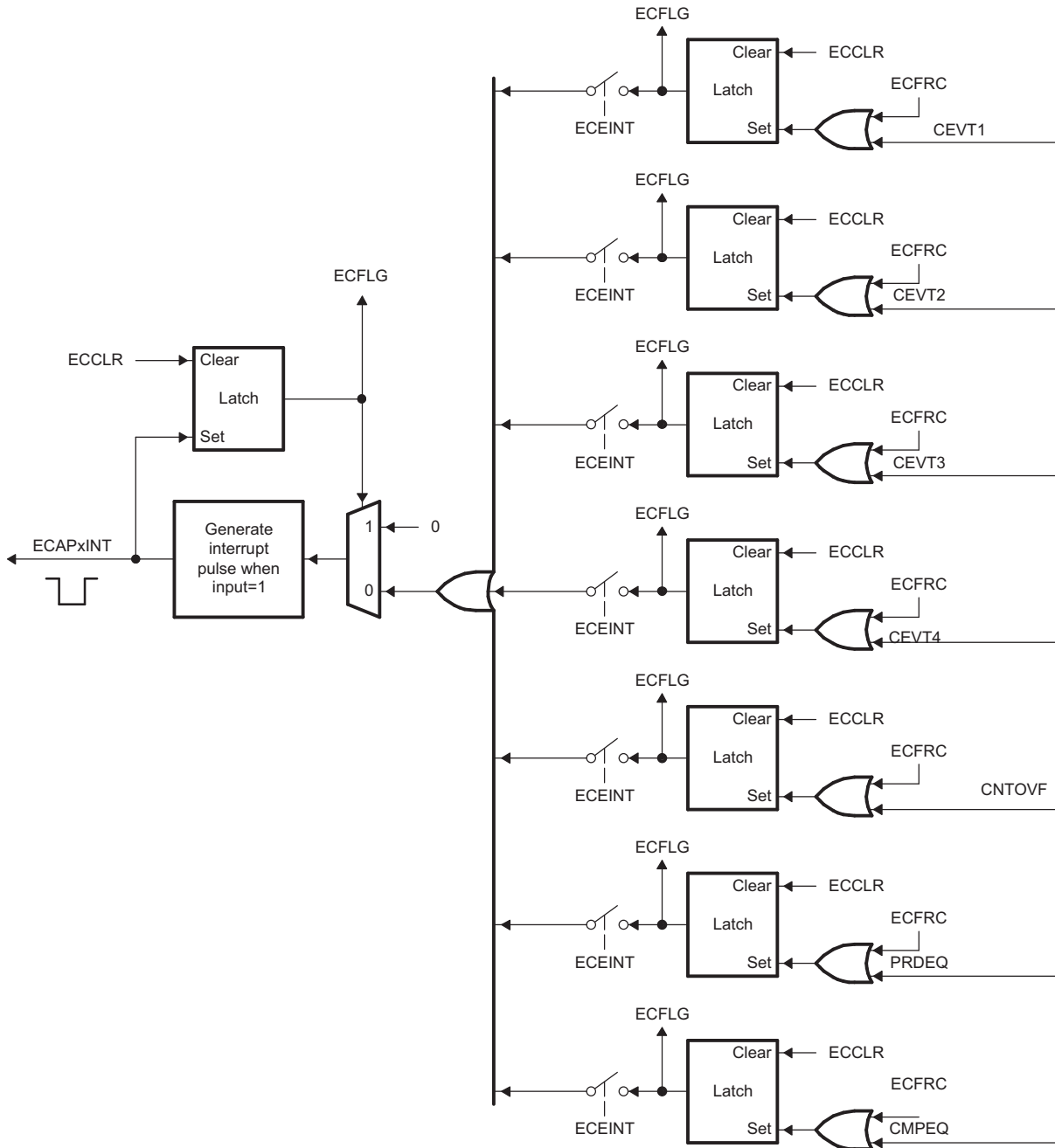
In capture mode, this logic inhibits (locks out) any shadow loading of CAP1 or CAP2 from APRD and ACMP registers, respectively.

In APWM mode, shadow loading is active and two choices are permitted:

- Immediate - APRD or ACMP are transferred to CAP1 or CAP2 immediately upon writing a new value.
- On period equal, CTR[31:0] = PRD[31:0]

NOTE: The CEVT1, CEVT2, CEVT3, CEVT4 flags are only active in capture mode (ECCTL2[CAP/APWM == 0]). The PRDEQ, CMPEQ flags are only valid in APWM mode (ECCTL2[CAP/APWM == 1]). CINTOVF flag is valid in both modes.

Figure 15-108. Interrupts in eCAP Module

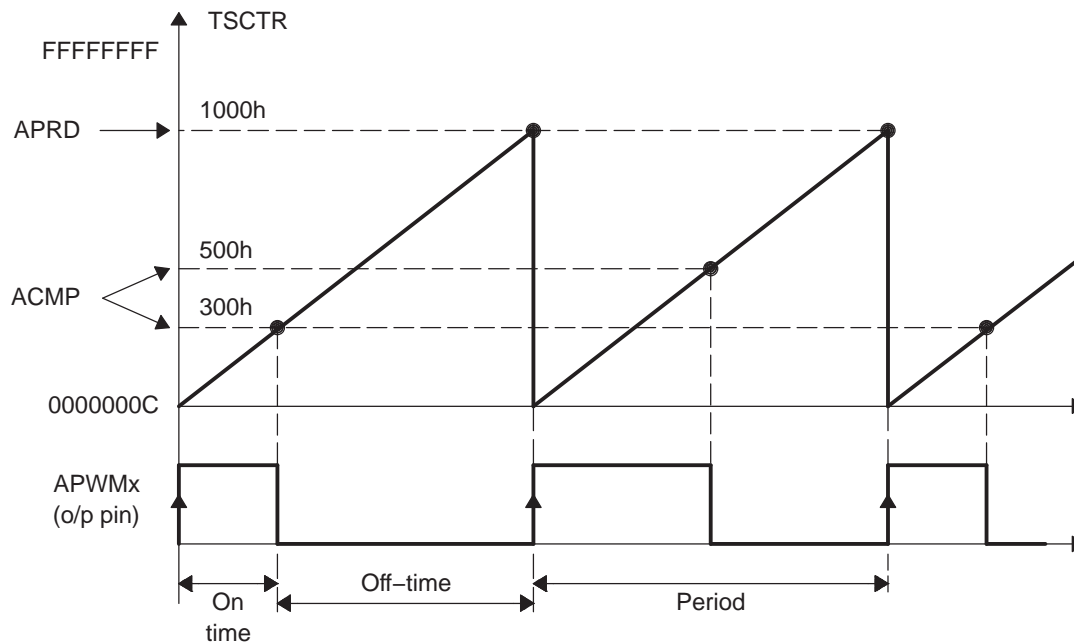


15.3.2.2.8 APWM Mode Operation

Main operating highlights of the APWM section:

- The time-stamp counter bus is made available for comparison via 2 digital (32-bit) comparators.
- When CAP1/2 registers are not used in capture mode, their contents can be used as Period and Compare values in APWM mode.
- Double buffering is achieved via shadow registers APRD and ACMP (CAP3/4). The shadow register contents are transferred over to CAP1/2 registers either immediately upon a write, or on a PRDEQ trigger.
- In APWM mode, writing to CAP1/CAP2 active registers will also write the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 will invoke the shadow mode.
- During initialization, you must write to the active registers for both period and compare. This automatically copies the initial values into the shadow values. For subsequent compare updates, during run-time, you only need to use the shadow registers.

Figure 15-109. PWM Waveform Details Of APWM Mode Operation



The behavior of APWM active-high mode (APWMPOL == 0) is:

CMP = 0x00000000, output low for duration of period (0% duty)

CMP = 0x00000001, output high 1 cycle

CMP = 0x00000002, output high 2 cycles

CMP = PERIOD, output high except for 1 cycle (<100% duty)

CMP = PERIOD+1, output high for complete period (100% duty)

CMP > PERIOD+1, output high for complete period

The behavior of APWM active-low mode (APWMPOL == 1) is:

CMP = 0x00000000, output high for duration of period (0% duty)

CMP = 0x00000001, output low 1 cycle

CMP = 0x00000002, output low 2 cycles

CMP = PERIOD, output low except for 1 cycle (<100% duty)

CMP = PERIOD+1, output low for complete period (100% duty)

CMP > PERIOD+1, output low for complete period

15.3.3 Use Cases

The following sections will provide Applications examples and code snippets to show how to configure and operate the eCAP module. For clarity and ease of use, below are useful #defines which will help in the understanding of the examples.

```

// ECCTL1 ( ECAP Control Reg 1)
//=====
// CAPxPOL bits
#define EC_RISING 0x0
#define EC_FALLING 0x1

// CTRRSTx bits
#define EC_ABS_MODE 0x0
#define EC_DELTA_MODE 0x1

// PRESCALE bits
#define EC_BYPASS 0x0
#define EC_DIV1 0x0
#define EC_DIV2 0x1
#define EC_DIV4 0x2
#define EC_DIV6 0x3
#define EC_DIV8 0x4
#define EC_DIV10 0x5

// ECCTL2 ( ECAP Control Reg 2)
//=====
// CONT/ONESHOT bit
#define EC_CONTINUOUS 0x0
#define EC_ONESHOT 0x1

// STOPVALUE bit
#define EC_EVENT1 0x0
#define EC_EVENT2 0x1
#define EC_EVENT3 0x2
#define EC_EVENT4 0x3

// RE-ARM bit
#define EC_ARM 0x1

// TSCTRSTOP bit
#define EC_FREEZE 0x0
#define EC_RUN 0x1

// SYNCO_SEL bit
#define EC_SYNCIN 0x0
#define EC_CTR_PRD 0x1
#define EC_SYNCO_DIS 0x2

// CAP/APWM mode bit
#define EC_CAP_MODE 0x0
#define EC_APWM_MODE 0x1

// APWMPOL bit
#define EC_ACTV_HI 0x0
#define EC_ACTV_LO 0x1

// Generic
#define EC_DISABLE 0x0
#define EC_ENABLE 0x1
#define EC_FORCE 0x1

```

15.3.3.1 Absolute Time-Stamp Operation Rising Edge Trigger Example

Figure 15-110 shows an example of continuous capture operation (Mod4 counter wraps around). In this figure, TSCTR counts-up without resetting and capture events are qualified on the rising edge only, this gives period (and frequency) information.

On an event, the TSCTR contents (time-stamp) is first captured, then Mod4 counter is incremented to the next state. When the TSCTR reaches FFFF FFFFh (maximum value), it wraps around to 0000 0000h (not shown in Figure 15-110), if this occurs, the CNTOVF (counter overflow) flag is set, and an interrupt (if enabled) occurs, CNTOVF (counter overflow) Flag is set, and an Interrupt (if enabled) occurs. Captured time-stamps are valid at the point indicated by the diagram, after the 4th event, hence event CEVT4 can conveniently be used to trigger an interrupt and the CPU can read data from the CAP n registers.

Figure 15-110. Capture Sequence for Absolute Time-Stamp, Rising Edge Detect

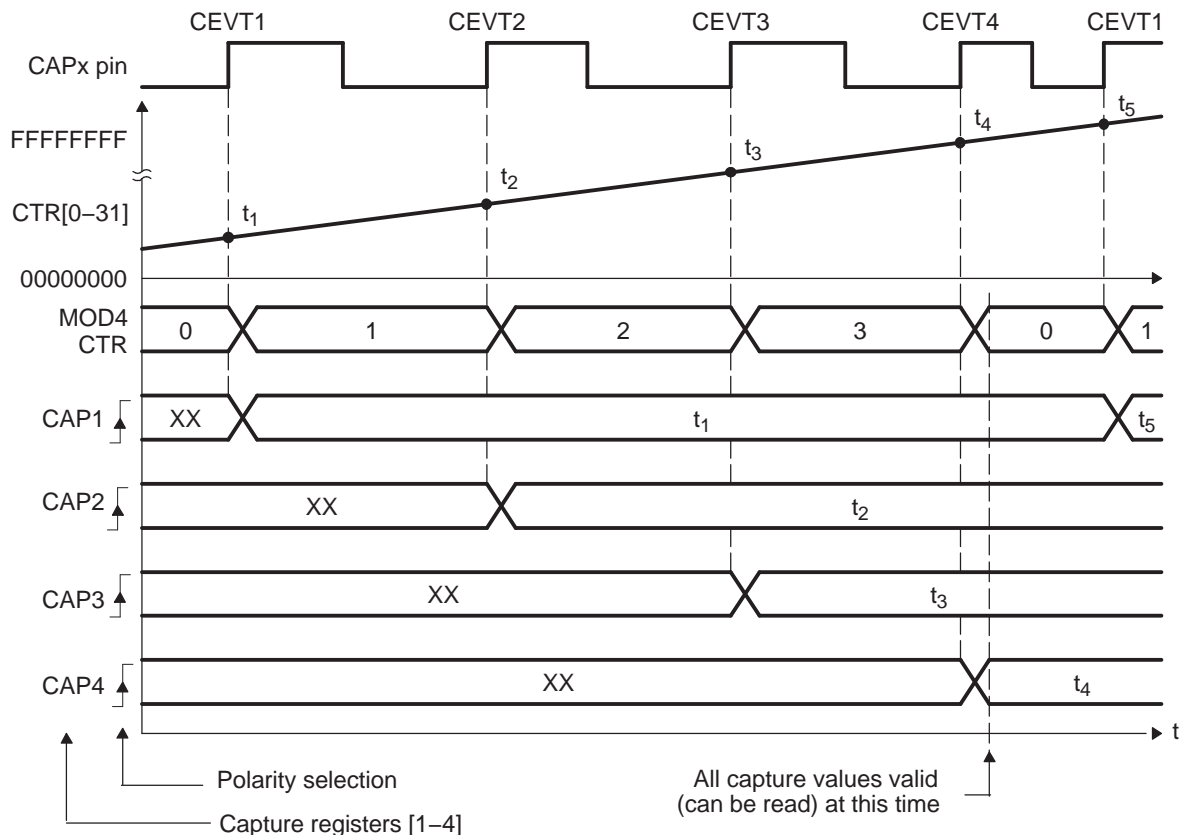


Table 15-94. ECAP Initialization for CAP Mode Absolute Time, Rising Edge Trigger

Register	Bit	Value
ECCTL1	CAP1POL	EC_RISING
ECCTL1	CAP2POL	EC_RISING
ECCTL1	CAP3POL	EC_RISING
ECCTL1	CAP4POL	EC_RISING
ECCTL1	CTRRST1	EC_ABS_MODE
ECCTL1	CTRRST2	EC_ABS_MODE
ECCTL1	CTRRST3	EC_ABS_MODE
ECCTL1	CTRRST4	EC_ABS_MODE
ECCTL1	CAPLDEN	EC_ENABLE
ECCTL1	PRESCALE	EC_DIV1
ECCTL2	CAP_APWM	EC_CAP_MODE
ECCTL2	CONT_ONESHT	EC_CONTINUOUS
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	SYNCL_EN	EC_DISABLE
ECCTL2	TSTRSTOP	EC_RUN

Example 15-18. Code Snippet for CAP Mode Absolute Time, Rising Edge Trigger

```
// Code snippet for CAP mode Absolute Time, Rising edge trigger

// Run Time ( e.g. CEVT4 triggered ISR call)
//=====
TSt1 = ECAPxRegs.CAP1;      // Fetch Time-Stamp captured at t1
TSt2 = ECAPxRegs.CAP2;      // Fetch Time-Stamp captured at t2
TSt3 = ECAPxRegs.CAP3;      // Fetch Time-Stamp captured at t3
TSt4 = ECAPxRegs.CAP4;      // Fetch Time-Stamp captured at t4

Period1 = TSt2-TSt1;        // Calculate 1st period
Period2 = TSt3-TSt2;        // Calculate 2nd period
Period3 = TSt4-TSt3;        // Calculate 3rd period
```

15.3.3.2 Absolute Time-Stamp Operation Rising and Falling Edge Trigger Example

In Figure 15-111 the eCAP operating mode is almost the same as in the previous section except capture events are qualified as either rising or falling edge, this now gives both period and duty cycle information: Period1 = $t_3 - t_1$, Period2 = $t_5 - t_3$, ...etc. Duty Cycle1 (on-time %) = $(t_2 - t_1) / \text{Period1} \times 100\%$, etc. Duty Cycle1 (off-time %) = $(t_3 - t_2) / \text{Period1} \times 100\%$, etc.

Figure 15-111. Capture Sequence for Absolute Time-Stamp, Rising and Falling Edge Detect

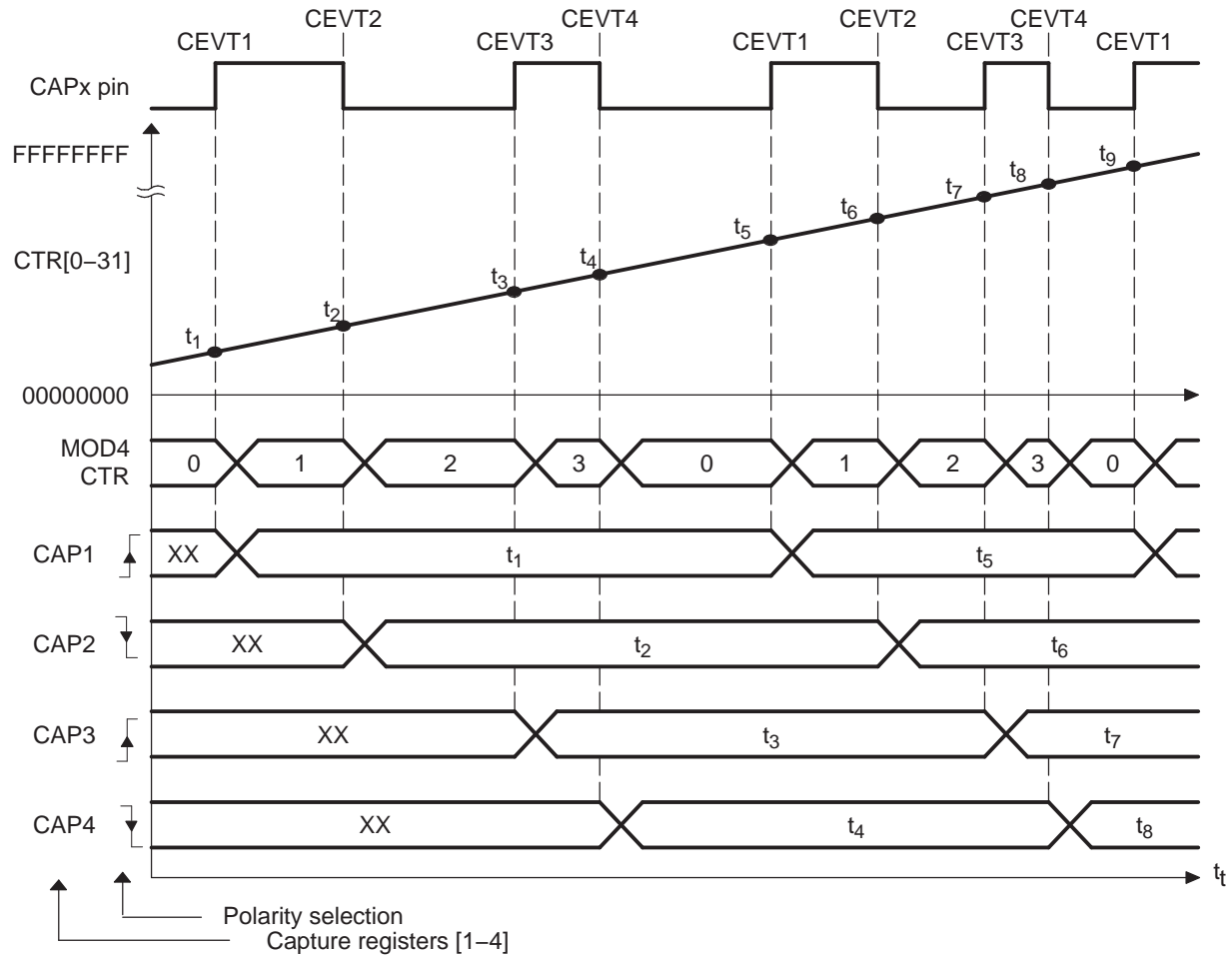


Table 15-95. ECAP Initialization for CAP Mode Absolute Time, Rising and Falling Edge Trigger

Register	Bit	Value
ECCTL1	CAP1POL	EC_RISING
ECCTL1	CAP2POL	EC_FALLING
ECCTL1	CAP3POL	EC_RISING
ECCTL1	CAP4POL	EC_FALLING
ECCTL1	CTRRST1	EC_ABS_MODE
ECCTL1	CTRRST2	EC_ABS_MODE
ECCTL1	CTRRST3	EC_ABS_MODE
ECCTL1	CTRRST4	EC_ABS_MODE
ECCTL1	CAPLDEN	EC_ENABLE
ECCTL1	PRESCALE	EC_DIV1
ECCTL2	CAP_APWM	EC_CAP_MODE
ECCTL2	CONT_ONESHT	EC_CONTINUOUS
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	SYNCL_EN	EC_DISABLE
ECCTL2	TSTRSTOP	EC_RUN

Example 15-19. Code Snippet for CAP Mode Absolute Time, Rising and Falling Edge Trigger

```
// Code snippet for CAP mode Absolute Time, Rising & Falling edge triggers

// Run Time ( e.g. CEVT4 triggered ISR call)
//=====
TSt1 = ECAPxRegs.CAP1;      // Fetch Time-Stamp captured at t1
TSt2 = ECAPxRegs.CAP2;      // Fetch Time-Stamp captured at t2
TSt3 = ECAPxRegs.CAP3;      // Fetch Time-Stamp captured at t3
TSt4 = ECAPxRegs.CAP4;      // Fetch Time-Stamp captured at t4

Period1 = TSt3-TSt1;        // Calculate 1st period
DutyOnTime1 = TSt2-TSt1;    // Calculate On time
DutyOffTime1 = TSt3-TSt2;   // Calculate Off time
```


15.3.3.3 Time Difference (Delta) Operation Rising Edge Trigger Example

Figure 15-112 shows how the eCAP module can be used to collect Delta timing data from pulse train waveforms. Here Continuous Capture mode (TSCTR counts-up without resetting, and Mod4 counter wraps around) is used. In Delta-time mode, TSCTR is Reset back to Zero on every valid event. Here Capture events are qualified as Rising edge only. On an event, TSCTR contents (time-stamp) is captured first, and then TSCTR is reset to Zero. The Mod4 counter then increments to the next state. If TSCTR reaches FFFF FFFFh (maximum value), before the next event, it wraps around to 0000 0000h and continues, a CNTOVF (counter overflow) Flag is set, and an Interrupt (if enabled) occurs. The advantage of Delta-time Mode is that the CAP n contents directly give timing data without the need for CPU calculations: Period1 = T_1 , Period2 = T_2 ,...etc. As shown in Figure 15-112, the CEVT1 event is a good trigger point to read the timing data, T_1 , T_2 , T_3 , T_4 are all valid here.

Figure 15-112. Capture Sequence for Delta Mode Time-Stamp, Rising Edge Detect

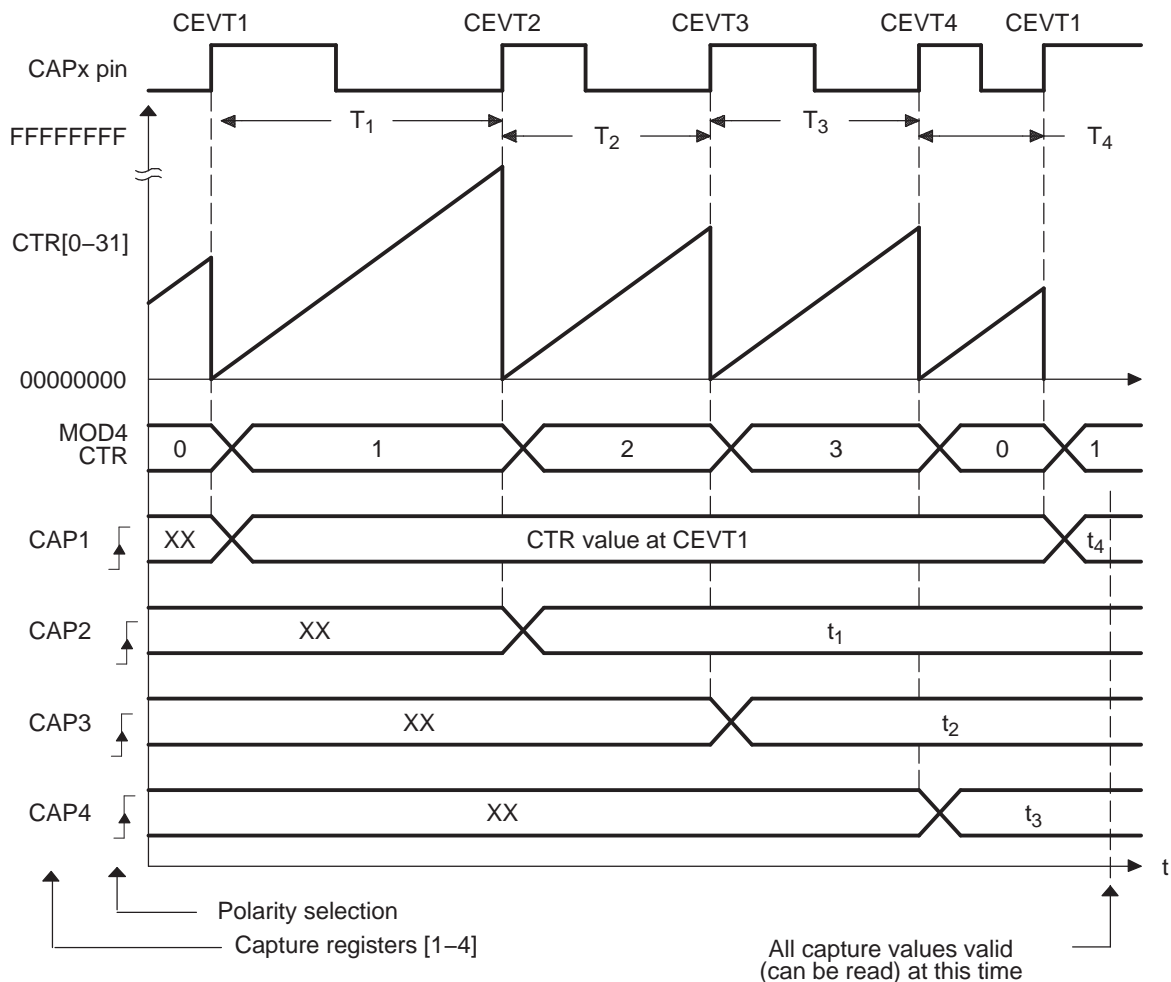


Table 15-96. ECAP Initialization for CAP Mode Delta Time, Rising Edge Trigger

Register	Bit	Value
ECCTL1	CAP1POL	EC_RISING
ECCTL1	CAP2POL	EC_RISING
ECCTL1	CAP3POL	EC_RISING
ECCTL1	CAP4POL	EC_RISING
ECCTL1	CTRRST1	EC_DELTA_MODE
ECCTL1	CTRRST2	EC_DELTA_MODE
ECCTL1	CTRRST3	EC_DELTA_MODE
ECCTL1	CTRRST4	EC_DELTA_MODE
ECCTL1	CAPLDEN	EC_ENABLE
ECCTL1	PRESCALE	EC_DIV1
ECCTL2	CAP_APWM	EC_CAP_MODE
ECCTL2	CONT_ONESHT	EC_CONTINUOUS
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	SYNCL_EN	EC_DISABLE
ECCTL2	TSTRSTOP	EC_RUN

Example 15-20. Code Snippet for CAP Mode Delta Time, Rising Edge Trigger

```
// Code snippet for CAP mode Delta Time, Rising edge trigger

// Run Time ( e.g. CEVT1 triggered ISR call)
//=====
// Note: here Time-stamp directly represents the Period value.
Period4 = ECAPxRegs.CAP1;    // Fetch Time-Stamp captured at T1
Period1 = ECAPxRegs.CAP2;    // Fetch Time-Stamp captured at T2
Period2 = ECAPxRegs.CAP3;    // Fetch Time-Stamp captured at T3
Period3 = ECAPxRegs.CAP4;    // Fetch Time-Stamp captured at T4
```

15.3.3.4 Time Difference (Delta) Operation Rising and Falling Edge Trigger Example

In Figure 15-113 the eCAP operating mode is almost the same as in previous section except Capture events are qualified as either Rising or Falling edge, this now gives both Period and Duty cycle information: $\text{Period1} = T_1 + T_2$, $\text{Period2} = T_3 + T_4$, ...etc $\text{Duty Cycle1 (on-time \%)} = T_1 / \text{Period1} \times 100\%$, etc $\text{Duty Cycle1 (off-time \%)} = T_2 / \text{Period1} \times 100\%$, etc

During initialization, you must write to the active registers for both period and compare. This will then automatically copy the init values into the shadow values. For subsequent compare updates, that is, during run-time, only the shadow registers must be used.

Figure 15-113. Capture Sequence for Delta Mode Time-Stamp, Rising and Falling Edge Detect

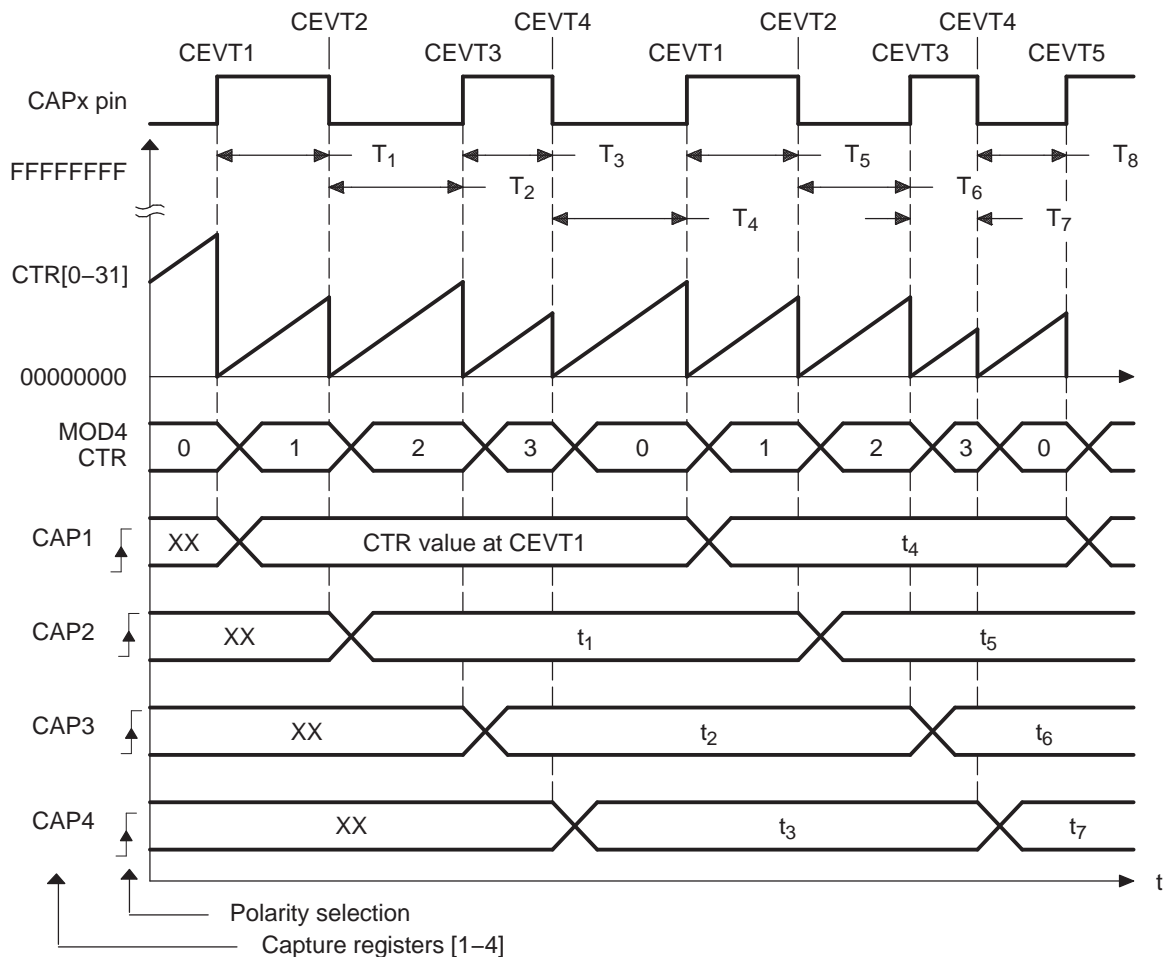


Table 15-97. ECAP Initialization for CAP Mode Delta Time, Rising and Falling Edge Triggers

Register	Bit	Value
ECCTL1	CAP1POL	EC_RISING
ECCTL1	CAP2POL	EC_FALLING
ECCTL1	CAP3POL	EC_RISING
ECCTL1	CAP4POL	EC_FALLING
ECCTL1	CTRRST1	EC_DELTA_MODE
ECCTL1	CTRRST2	EC_DELTA_MODE
ECCTL1	CTRRST3	EC_DELTA_MODE
ECCTL1	CTRRST4	EC_DELTA_MODE
ECCTL1	CAPLDEN	EC_ENABLE
ECCTL1	PRESCALE	EC_DIV1
ECCTL2	CAP_APWM	EC_CAP_MODE
ECCTL2	CONT_ONESHT	EC_CONTINUOUS
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	SYNCL_EN	EC_DISABLE
ECCTL2	TSTRSTOP	EC_RUN

Example 15-21. Code Snippet for CAP Mode Delta Time, Rising and Falling Edge Triggers

```
// Code snippet for CAP mode Delta Time, Rising and Falling edge triggers

// Run Time ( e.g. CEVT1 triggered ISR call)
//=====
// Note: here Time-stamp directly represents the Duty cycle values.
DutyOnTime1 = ECAPxRegs.CAP2;    // Fetch Time-Stamp captured at T2
DutyOffTime1 = ECAPxRegs.CAP3;   // Fetch Time-Stamp captured at T3
DutyOnTime2 = ECAPxRegs.CAP4;    // Fetch Time-Stamp captured at T4
DutyOffTime2 = ECAPxRegs.CAP1;   // Fetch Time-Stamp captured at T1

Period1 = DutyOnTime1 + DutyOffTime1;
Period2 = DutyOnTime2 + DutyOffTime2;
```

15.3.3.5 Application of the APWM Mode

15.3.3.5.1 Simple PWM Generation (Independent Channel/s) Example

In this example, the eCAP module is configured to operate as a PWM generator. Here a very simple single channel PWM waveform is generated from output pin APWM n . The PWM polarity is active high, which means that the compare value (CAP2 reg is now a compare register) represents the on-time (high level) of the period. Alternatively, if the APWMPOL bit is configured for active low, then the compare value represents the off-time.

Figure 15-114. PWM Waveform Details of APWM Mode Operation

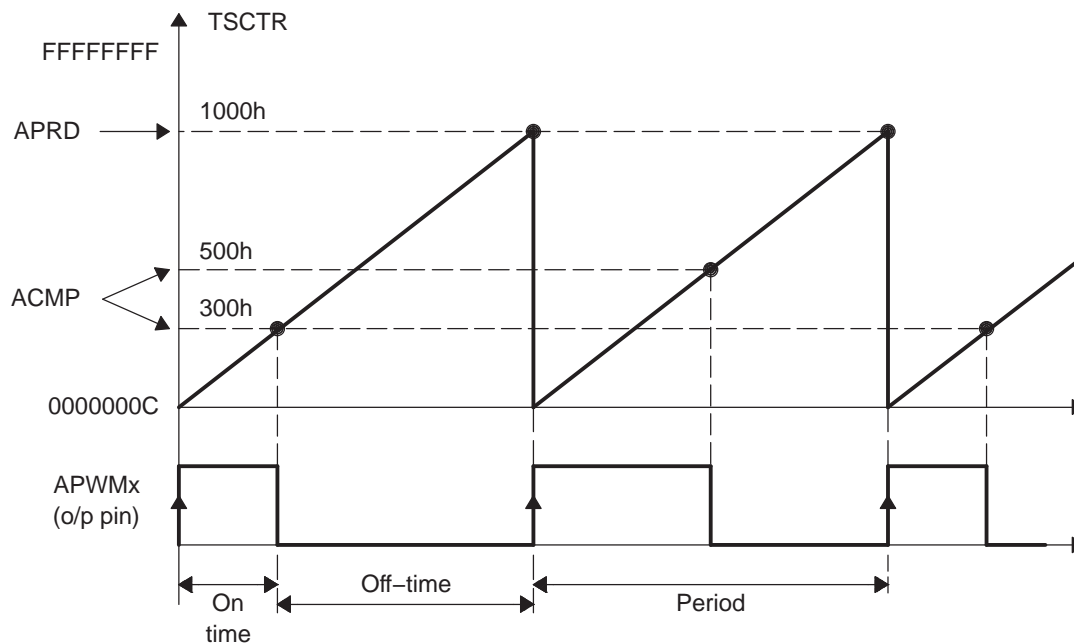


Table 15-98. ECAP Initialization for APWM Mode

Register	Bit	Value
CAP1	CAP1	0x1000
CTRPHS	CTRPHS	0x0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCL_EN	EC_DISABLE
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	TSTRSTOP	EC_RUN

Example 15-22. Code Snippet for APWM Mode

```

// Code snippet for APWM mode Example 1

// Run Time (Instant 1, e.g. ISR call)
//=====
    ECAPxRegs.CAP2 = 0x300;      // Set Duty cycle i.e. compare value

// Run Time (Instant 2, e.g. another ISR call)
//=====
    ECAPxRegs.CAP2 = 0x500;      // Set Duty cycle i.e. compare value

```

15.3.3.5.2 Multichannel PWM Generation with Synchronization Example

Figure 15-115 takes advantage of the synchronization feature between eCAP modules. Here 4 independent PWM channels are required with different frequencies, but at integer multiples of each other to avoid "beat" frequencies. Hence one eCAP module is configured as the Master and the remaining 3 are Slaves all receiving their synch pulse (CTR = PRD) from the master. Note the Master is chosen to have the lower frequency ($F_1 = 1/20,000$) requirement. Here Slave2 Freq = $2 \times F_1$, Slave3 Freq = $4 \times F_1$ and Slave4 Freq = $5 \times F_1$. Note here values are in decimal notation. Also, only the APWM1 output waveform is shown.

Figure 15-115. Multichannel PWM Example Using 4 eCAP Modules

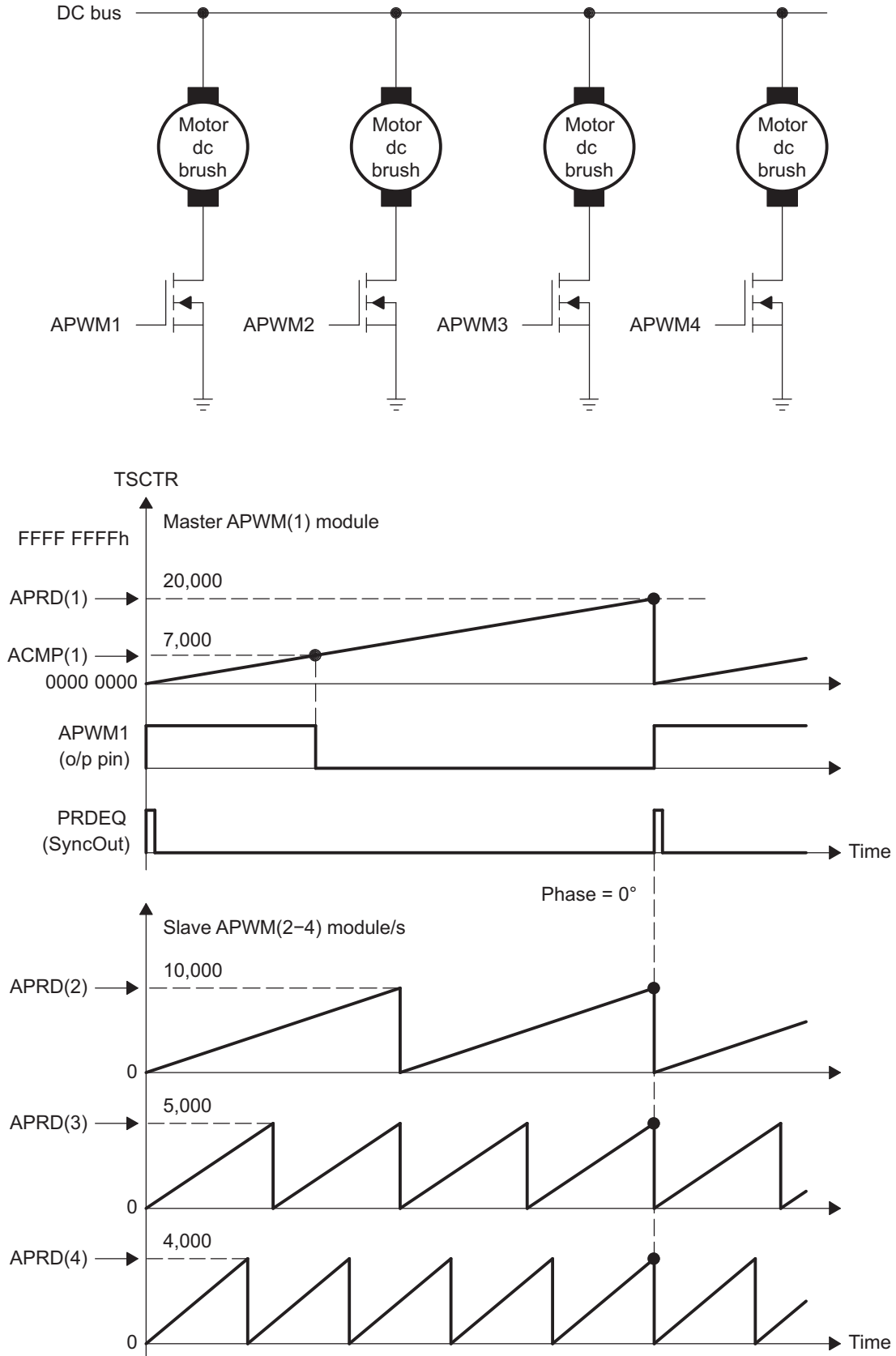


Table 15-99. ECAP1 Initialization for Multichannel PWM Generation with Synchronization

Register	Bit	Value
CAP1	CAP1	20000
CTRPHS	CTRPHS	0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCL_EN	EC_DISABLE
ECCTL2	SYNCO_SEL	EC_CTR_PRD
ECCTL2	TSCTRSTOP	EC_RUN

Table 15-100. ECAP2 Initialization for Multichannel PWM Generation with Synchronization

Register	Bit	Value
CAP1	CAP1	10000
CTRPHS	CTRPHS	0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCL_EN	EC_ENABLE
ECCTL2	SYNCO_SEL	EC_SYNCI
ECCTL2	TSCTRSTOP	EC_RUN

Table 15-101. ECAP3 Initialization for Multichannel PWM Generation with Synchronization

Register	Bit	Value
CAP1	CAP1	5000
CTRPHS	CTRPHS	0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCL_EN	EC_ENABLE
ECCTL2	SYNCO_SEL	EC_SYNCI
ECCTL2	TSCTRSTOP	EC_RUN

Table 15-102. ECAP4 Initialization for Multichannel PWM Generation with Synchronization

Register	Bit	Value
CAP1	CAP1	4000
CTRPHS	CTRPHS	0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCL_EN	EC_ENABLE
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	TSCTRSTOP	EC_RUN

Example 15-23. Code Snippet for Multichannel PWM Generation with Synchronization

```

// Code snippet for APWM mode Example 2

// Run Time (Note: Example execution of one run-time instant)
//=====
ECAP1Regs.CAP2 = 7000;    // Set Duty cycle i.e., compare value = 7000
ECAP2Regs.CAP2 = 2000;    // Set Duty cycle i.e., compare value = 2000
ECAP3Regs.CAP2 = 550;     // Set Duty cycle i.e., compare value = 550
ECAP4Regs.CAP2 = 6500;    // Set Duty cycle i.e., compare value = 6500
    
```

15.3.3.5.3 Multichannel PWM Generation with Phase Control Example

In [Figure 15-116](#), the Phase control feature of the APWM mode is used to control a 3 phase Interleaved DC/DC converter topology. This topology requires each phase to be off-set by 120° from each other. Hence if “Leg” 1 (controlled by APWM1) is the reference Leg (or phase), that is, 0°, then Leg 2 need 120° off-set and Leg 3 needs 240° off-set. The waveforms in [Figure 15-116](#) show the timing relationship between each of the phases (Legs). Note eCAP1 module is the Master and issues a sync out pulse to the slaves (modules 2, 3) whenever TSCTR = Period value.

Figure 15-116. Multiphase (channel) Interleaved PWM Example Using 3 eCAP Modules

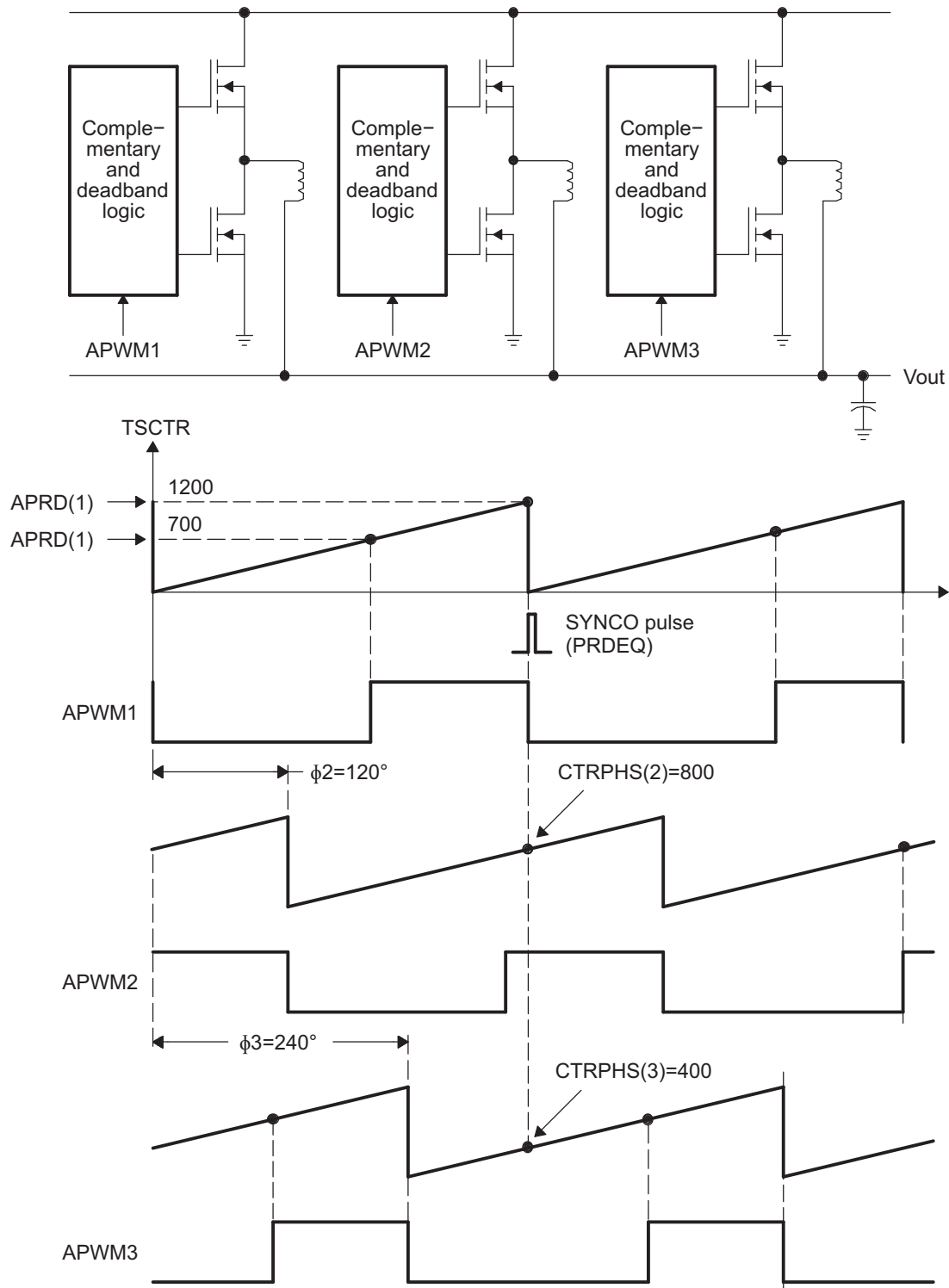


Table 15-103. ECAP1 Initialization for Multichannel PWM Generation with Phase Control

Register	Bit	Value
CAP1	CAP1	1200
CTRPHS	CTRPHS	0
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCL_EN	EC_DISABLE
ECCTL2	SYNCO_SEL	EC_CTR_PRD
ECCTL2	TSCTRSTOP	EC_RUN

Table 15-104. ECAP2 Initialization for Multichannel PWM Generation with Phase Control

Register	Bit	Value
CAP1	CAP1	1200
CTRPHS	CTRPHS	800
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCL_EN	EC_ENABLE
ECCTL2	SYNCO_SEL	EC_SYNCL
ECCTL2	TSCTRSTOP	EC_RUN

Table 15-105. ECAP3 Initialization for Multichannel PWM Generation with Phase Control

Register	Bit	Value
CAP1	CAP1	1200
CTRPHS	CTRPHS	400
ECCTL2	CAP_APWM	EC_APWM_MODE
ECCTL2	APWMPOL	EC_ACTV_HI
ECCTL2	SYNCL_EN	EC_ENABLE
ECCTL2	SYNCO_SEL	EC_SYNCO_DIS
ECCTL2	TSCTRSTOP	EC_RUN

Example 15-24. Code Snippet for Multichannel PWM Generation with Phase Control

```
// Code snippet for APWM mode Example 3

// Run Time (Note: Example execution of one run-time instant)
//=====
// All phases are set to the same duty cycle
ECAP1Regs.CAP2 = 700;    // Set Duty cycle i.e. compare value = 700
ECAP2Regs.CAP2 = 700;    // Set Duty cycle i.e. compare value = 700
ECAP3Regs.CAP2 = 700;    // Set Duty cycle i.e. compare value = 700
```

15.3.4 Registers

All 32-bit registers are aligned on even address boundaries and are organized in little-endian mode. The 16 least-significant bits of a 32-bit register are located on lowest address (even address).

NOTE: In APWM mode, writing to CAP1/CAP2 active registers also writes the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 invokes the shadow mode.

15.3.4.1 ECAP Registers

[Table 15-106](#) lists the memory-mapped registers for the ECAP. All register offset addresses not listed in [Table 15-106](#) should be considered as reserved locations and the register contents should not be modified.

Table 15-106. ECAP Registers

Offset	Acronym	Register Name	Section
0h	TSCTR	Time-Stamp Counter Register	Section 15.3.4.1.1
4h	CTRPHS	Counter Phase Offset Value Register	Section 15.3.4.1.2
8h	CAP1	Capture 1 Register	Section 15.3.4.1.3
Ch	CAP2	Capture 2 Register	Section 15.3.4.1.4
10h	CAP3	Capture 3 Register	Section 15.3.4.1.5
14h	CAP4	Capture 4 Register	Section 15.3.4.1.6
28h	ECCTL1	Capture Control Register 1	Section 15.3.4.1.7
2Ah	ECCTL2	Capture Control Register 2	Section 15.3.4.1.8
2Ch	ECEINT	Capture Interrupt Enable Register	Section 15.3.4.1.9
2Eh	ECFLG	Capture Interrupt Flag Register	Section 15.3.4.1.10
30h	ECCLR	Capture Interrupt Clear Register	Section 15.3.4.1.11
32h	ECFRC	Capture Interrupt Force Register	Section 15.3.4.1.12
5Ch	REVID	Revision ID Register	Section 15.4.3.25

15.3.4.1.1 TSCTR Register (offset = 0h) [reset = 0h]

TSCTR is shown in [Figure 15-117](#) and described in [Table 15-107](#).

Figure 15-117. TSCTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSCTR																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-107. TSCTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSCTR	R/W	0h	Active 32 bit counter register that is used as the capture time-base

15.3.4.1.2 CTRPHS Register (offset = 4h) [reset = 0h]

CTRPHS is shown in [Figure 15-118](#) and described in [Table 15-108](#).

Figure 15-118. CTRPHS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTRPHS																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-108. CTRPHS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CTRPHS	R/W	0h	Counter phase value register that can be programmed for phase lag/lead. This register shadows TSCTR and is loaded into TSCTR upon either a SYNCI event or S/W force via a control bit. Used to achieve phase control synchronization with respect to other eCAP and EPWM time-bases.

15.3.4.1.3 CAP1 Register (offset = 8h) [reset = 0h]

CAP1 is shown in [Figure 15-119](#) and described in [Table 15-109](#).

Figure 15-119. CAP1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP1																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-109. CAP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP1	R/W	0h	This register can be loaded (written) by the following. (a) Time-Stamp (that is, counter value) during a capture event. (b) Software may be useful for test purposes. (c) APRD active register when used in APWM mode.

15.3.4.1.4 CAP2 Register (offset = Ch) [reset = 0h]

CAP2 is shown in [Figure 15-120](#) and described in [Table 15-110](#).

Figure 15-120. CAP2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP2																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-110. CAP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP2	R/W	0h	This register can be loaded (written) by the following. (a) Time-Stamp (that is, counter value) during a capture event. (b) Software may be useful for test purposes. (c) ACMP active register when used in APWM mode.

15.3.4.1.5 CAP3 Register (offset = 10h) [reset = 0h]

CAP3 is shown in [Figure 15-121](#) and described in [Table 15-111](#).

Figure 15-121. CAP3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP3																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-111. CAP3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP3	R/W	0h	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the period shadow (APRD) register. You update the PWM period value through this register. In this mode, CAP3 shadows CAP1.

15.3.4.1.6 CAP4 Register (offset = 14h) [reset = 0h]

CAP4 is shown in [Figure 15-122](#) and described in [Table 15-112](#).

Figure 15-122. CAP4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP4																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-112. CAP4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP4	R/W	0h	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the compare shadow (ACMP) register. You update the PWM compare value through this register. In this mode, CAP4 shadows CAP2.

15.3.4.1.7 ECCTL1 Register (offset = 28h) [reset = 0h]

 ECCTL1 is shown in [Figure 15-123](#) and described in [Table 15-113](#).

Figure 15-123. ECCTL1 Register

15	14	13	12	11	10	9	8
FREE_SOFT		PRESCALE				CAPLDEN	
R/W-0h		R/W-0h				R/W-0h	
7	6	5	4	3	2	1	0
CTRRST4	CAP4POL	CTRRST3	CAP3POL	CTRRST2	CAP2POL	CTRRST1	CAP1POL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-113. ECCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation Control 0h = TSCTR counter stops immediately on emulation suspend. 1h = TSCTR counter runs until = 0. 2h = TSCTR counter is unaffected by emulation suspend (Run Free). 3h = TSCTR counter is unaffected by emulation suspend (Run Free).
13-9	PRESCALE	R/W	0h	Event Filter prescale select ... 0h = Divide by 1 (i.e., no prescale, by-pass the prescaler) 1h = Divide by 2 2h = Divide by 4 3h = Divide by 6 4h = Divide by 8 5h = Divide by 10 1Eh = Divide by 60 1Fh = Divide by 62
8	CAPLDEN	R/W	0h	Enable Loading of CAP1 to CAP4 registers on a capture event 0h = Disable CAP1 through 4 register loads at capture event time. 1h = Enable CAP1-4 register loads at capture event time.
7	CTRRST4	R/W	0h	Counter Reset on Capture Event 4 0h = Do not reset counter on Capture Event 4 (absolute time stamp operation) 1h = Reset counter after Capture Event 4 time-stamp has been captured (used in difference mode operation)
6	CAP4POL	R/W	0h	Capture Event 4 Polarity select 0h = Capture Event 4 triggered on a rising edge (RE) 1h = Capture Event 4 triggered on a falling edge (FE)
5	CTRRST3	R/W	0h	Counter Reset on Capture Event 3 0h = Do not reset counter on Capture Event 3 (absolute time stamp) 1h = Reset counter after Event 3 time-stamp has been captured (used in difference mode operation)
4	CAP3POL	R/W	0h	Capture Event 3 Polarity select 0h = Capture Event 3 triggered on a rising edge (RE) 1h = Capture Event 3 triggered on a falling edge (FE)
3	CTRRST2	R/W	0h	Counter Reset on Capture Event 2 0h = Do not reset counter on Capture Event 2 (absolute time stamp) 1h = Reset counter after Event 2 time-stamp has been captured (used in difference mode operation)

Table 15-113. ECCTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	CAP2POL	R/W	0h	Capture Event 2 Polarity select 0h = Capture Event 2 triggered on a rising edge (RE) 1h = Capture Event 2 triggered on a falling edge (FE)
1	CTRRST1	R/W	0h	Counter Reset on Capture Event 1 0h = Do not reset counter on Capture Event 1 (absolute time stamp) 1h = Reset counter after Event 1 time-stamp has been captured (used in difference mode operation)
0	CAP1POL	R/W	0h	Capture Event 1 Polarity select 0h = Capture Event 1 triggered on a rising edge (RE) 1h = Capture Event 1 triggered on a falling edge (FE)

15.3.4.1.8 ECCTL2 Register (offset = 2Ah) [reset = 6h]

 ECCTL2 is shown in [Figure 15-124](#) and described in [Table 15-114](#).

Figure 15-124. ECCTL2 Register

15	14	13	12	11	10	9	8
RESERVED					APWMPOL	CAP_APWM	SWSYNC
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
SYNCO_SEL		SYNCL_EN	TSCTRSTOP	RE-ARM	STOP_WRAP		CONT_ONESHOT
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-3h		R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-114. ECCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	
10	APWMPOL	R/W	0h	APWM output polarity select. This is applicable only in APWM operating mode 0h = Output is active high (Compare value defines high time) 1h = Output is active low (Compare value defines low time)
9	CAP_APWM	R/W	0h	CAP/APWM operating mode select 0h = ECAP module operates in capture mode. This mode forces the following configuration. (a) Inhibits TSCTR resets via PRDEQ event. (b) Inhibits shadow loads on CAP1 and 2 registers. (c) Permits user to enable CAP1-4 register load. (d) ECAPn/APWMn pin operates as a capture input. 1h = ECAP module operates in APWM mode. This mode forces the following configuration. (a) Resets TSCTR on PRDEQ event (period boundary). (b) Permits shadow loading on CAP1 and 2 registers. (c) Disables loading of time-stamps into CAP1-4 registers. (d) ECAPn/APWMn pin operates as a APWM output.
8	SWSYNC	R/W	0h	Software-forced Counter (TSCTR) Synchronizing. This provides a convenient software method to synchronize some or all ECAP time bases. In APWM mode, the synchronizing can also be done via the PRDEQ event. Note: Selecting PRDEQ is meaningful only in APWM mode. However, you can choose it in CAP mode if you find doing so useful. 0h = Writing a zero has no effect. Reading always returns a zero 1h = Writing a one forces a TSCTR shadow load of current ECAP module and any ECAP modules down-stream providing the SYNCO_SEL bits are 0,0. After writing a 1, this bit returns to a zero.
7-6	SYNCO_SEL	R/W	0h	Sync-Out Select 0h = Select sync-in event to be the sync-out signal (pass through) 1h = Select PRDEQ event to be the sync-out signal 2h = Disable sync out signal 3h = Disable sync out signal
5	SYNCL_EN	R/W	0h	Counter (TSCTR) Sync-In select mode 0h = Disable sync-in option 1h = Enable counter (TSCTR) to be loaded from CTRPHS register upon either a SYNCL signal or a S/W force event.
4	TSCTRSTOP	R/W	0h	Time Stamp (TSCTR) Counter Stop (freeze) Control 0h = TSCTR stopped 1h = TSCTR free-running

Table 15-114. ECCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	RE-ARM	R/W	0h	<p>One-Shot Re-Arming Control, that is, wait for stop trigger.</p> <p>Note: The re-arm function is valid in one shot or continuous mode.</p> <p>0h = Has no effect (reading always returns a 0)</p> <p>1h = Arms the one-shot sequence as follows: 1) Resets the Mod4 counter to zero. 2) Unfreezes the Mod4 counter. 3) Enables capture register loads.</p>
2-1	STOP_WRAP	R/W	3h	<p>Stop value for one-shot mode.</p> <p>This is the number (between 1 and 4) of captures allowed to occur before the CAP (1 through 4) registers are frozen, that is, capture sequence is stopped.</p> <p>Wrap value for continuous mode.</p> <p>This is the number (between 1 and 4) of the capture register in which the circular buffer wraps around and starts again.</p> <p>Notes: STOP_WRAP is compared to Mod4 counter and, when equal, the following two actions occur.</p> <p>(1) Mod4 counter is stopped (frozen), and (2) Capture register loads are inhibited.</p> <p>In one-shot mode, further interrupt events are blocked until re-armed.</p> <p>0h = Stop after Capture Event 1 in one-shot mode. Wrap after Capture Event 1 in continuous mode.</p> <p>1h = Stop after Capture Event 2 in one-shot mode. Wrap after Capture Event 2 in continuous mode.</p> <p>2h = Stop after Capture Event 3 in one-shot mode. Wrap after Capture Event 3 in continuous mode.</p> <p>3h = Stop after Capture Event 4 in one-shot mode. Wrap after Capture Event 4 in continuous mode.</p>
0	CONT_ONESHT	R/W	0h	<p>Continuous or one-shot mode control (applicable only in capture mode)</p> <p>0h = Operate in continuous mode</p> <p>1h = Operate in one-shot mode</p>

15.3.4.1.9 ECEINT Register (offset = 2Ch) [reset = 0h]

ECEINT is shown in [Figure 15-125](#) and described in [Table 15-115](#).

The interrupt enable bits (CEVTn) block any of the selected events from generating an interrupt. Events will still be latched into the flag bit (ECFLG register) and can be forced or cleared via the ECFRC and ECCLR registers. The proper procedure for configuring peripheral modes and interrupts is: 1. Disable global interrupts. 2. Stop eCAP counter. 3. Disable eCAP interrupts. 4. Configure peripheral registers. 5. Clear spurious eCAP interrupt flags. 6. Enable eCAP interrupts. 7. Start eCAP counter. 8. Enable global interrupts.

Figure 15-125. ECEINT Register

15								14								13								12								11								10								9								8							
RESERVED																																																															
R-0h																																																															
7								6								5								4								3								2								1								0							
CMPEQ								PRDEQ								CNTOVF								CEVT4								CEVT3								CEVT2								CEVT1								RESERVED							
R/W-0h								R/W-0h								R/W-0h								R/W-0h								R/W-0h								R/W-0h								R/W-0h								R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-115. ECEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	
7	CMPEQ	R/W	0h	Counter Equal Compare Interrupt Enable. 0h = Disable Compare Equal as an Interrupt source. 1h = Enable Compare Equal as an Interrupt source.
6	PRDEQ	R/W	0h	Counter Equal Period Interrupt Enable. 0h = Disable Period Equal as an Interrupt source. 1h = Enable Period Equal as an Interrupt source.
5	CNTOVF	R/W	0h	Counter Overflow Interrupt Enable. 0h = Disable counter Overflow as an Interrupt source. 1h = Enable counter Overflow as an Interrupt source.
4	CEVT4	R/W	0h	Capture Event 4 Interrupt Enable. 0h = Disable Capture Event 4 as an Interrupt source. 1h = Enable Capture Event 4 as an Interrupt source.
3	CEVT3	R/W	0h	Capture Event 3 Interrupt Enable. 0h = Disable Capture Event 3 as an Interrupt source. 1h = Enable Capture Event 3 as an Interrupt source.
2	CEVT2	R/W	0h	Capture Event 2 Interrupt Enable. 0h = Disable Capture Event 2 as an Interrupt source. 1h = Enable Capture Event 2 as an Interrupt source.
1	CEVT1	R/W	0h	Capture Event 1 Interrupt Enable . 0h = Disable Capture Event 1 as an Interrupt source. 1h = Enable Capture Event 1 as an Interrupt source.
0	RESERVED	R	0h	

15.3.4.1.10 ECFLG Register (offset = 2Eh) [reset = 0h]

 ECFLG is shown in [Figure 15-126](#) and described in [Table 15-116](#).

Figure 15-126. ECFLG Register

15		14		13		12		11		10		9		8	
RESERVED															
R-0h															
7		6		5		4		3		2		1		0	
CMPEQ	PRDEQ	CNTOVF	CEVT4	CEVT3	CEVT2	CEVT1	INT								
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h								

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-116. ECFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	
7	CMPEQ	R	0h	Compare Equal Compare Status Flag. This flag is only active in APWM mode. 0h = Indicates no event occurred 1h = Indicates the counter (TSCTR) reached the compare register value (ACMP)
6	PRDEQ	R	0h	Counter Equal Period Status Flag. This flag is only active in APWM mode. 0h = Indicates no event occurred 1h = Indicates the counter (TSCTR) reached the period register value (APRD) and was reset.
5	CNTOVF	R	0h	Counter Overflow Status Flag. This flag is active in CAP and APWM mode. 0h = Indicates no event occurred. 1h = Indicates the counter (TSCTR) has made the transition from 0xFFFFFFFF to 0x00000000
4	CEVT4	R	0h	Capture Event 4 Status Flag This flag is only active in CAP mode. 0h = Indicates no event occurred 1h = Indicates the fourth event occurred at ECAPn pin
3	CEVT3	R	0h	Capture Event 3 Status Flag. This flag is active only in CAP mode. 0h = Indicates no event occurred. 1h = Indicates the third event occurred at ECAPn pin.
2	CEVT2	R	0h	Capture Event 2 Status Flag. This flag is only active in CAP mode. 0h = Indicates no event occurred. 1h = Indicates the second event occurred at ECAPn pin.
1	CEVT1	R	0h	Capture Event 1 Status Flag. This flag is only active in CAP mode. 0h = Indicates no event occurred. 1h = Indicates the first event occurred at ECAPn pin.
0	INT	R	0h	Global Interrupt Status Flag 0h = Indicates no interrupt generated. 1h = Indicates that an interrupt was generated.

15.3.4.1.11 ECCLR Register (offset = 30h) [reset = 0h]

 ECCLR is shown in [Figure 15-127](#) and described in [Table 15-117](#).

Figure 15-127. ECCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CMPEQ	PRDEQ	CNTOVF	CEVT4	CEVT3	CEVT2	CEVT1	INT
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-117. ECCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	
7	CMPEQ	R/W	0h	Counter Equal Compare Status Flag 0h = Writing a 0 has no effect. Always reads back a 0 1h = Writing a 1 clears the CMPEQ flag condition
6	PRDEQ	R/W	0h	Counter Equal Period Status Flag 0h = Writing a 0 has no effect. Always reads back a 0 1h = Writing a 1 clears the PRDEQ flag condition
5	CNTOVF	R/W	0h	Counter Overflow Status Flag 0h = Writing a 0 has no effect. Always reads back a 0 1h = Writing a 1 clears the CNTOVF flag condition
4	CEVT4	R/W	0h	Capture Event 4 Status Flag 0h = Writing a 0 has no effect. Always reads back a 0. 1h = Writing a 1 clears the CEVT3 flag condition.
3	CEVT3	R/W	0h	Capture Event 3 Status Flag 0h = Writing a 0 has no effect. Always reads back a 0. 1h = Writing a 1 clears the CEVT3 flag condition.
2	CEVT2	R/W	0h	Capture Event 2 Status Flag 0h = Writing a 0 has no effect. Always reads back a 0. 1h = Writing a 1 clears the CEVT2 flag condition.
1	CEVT1	R/W	0h	Capture Event 1 Status Flag 0h = Writing a 0 has no effect. Always reads back a 0. 1h = Writing a 1 clears the CEVT1 flag condition.
0	INT	R/W	0h	Global Interrupt Clear Flag 0h = Writing a 0 has no effect. Always reads back a 0. 1h = Writing a 1 clears the INT flag and enable further interrupts to be generated if any of the event flags are set to 1.

15.3.4.1.12 ECFRC Register (offset = 32h) [reset = 0h]

ECFRC is shown in [Figure 15-128](#) and described in [Table 15-118](#).

Figure 15-128. ECFRC Register

15		14		13		12		11		10		9		8	
RESERVED															
R-0h															
7		6		5		4		3		2		1		0	
CMPEQ		PRDEQ		CNTOVF		CEVT4		CEVT3		CEVT2		CEVT1		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-118. ECFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	
7	CMPEQ	R/W	0h	Force Counter Equal Compare Interrupt 0h = No effect. Always reads back a 0. 1h = Writing a 1 sets the CMPEQ flag bit.
6	PRDEQ	R/W	0h	Force Counter Equal Period Interrupt 0h = No effect. Always reads back a 0. 1h = Writing a 1 sets the PRDEQ flag bit.
5	CNTOVF	R/W	0h	Force Counter Overflow 0h = No effect. Always reads back a 0. 1h = Writing a 1 to this bit sets the CNTOVF flag bit.
4	CEVT4	R/W	0h	Force Capture Event 4 0h = No effect. Always reads back a 0. 1h = Writing a 1 sets the CEVT4 flag bit
3	CEVT3	R/W	0h	Force Capture Event 3 0h = No effect. Always reads back a 0. 1h = Writing a 1 sets the CEVT3 flag bit
2	CEVT2	R/W	0h	Force Capture Event 2 0h = No effect. Always reads back a 0. 1h = Writing a 1 sets the CEVT2 flag bit.
1	CEVT1	R/W	0h	Always reads back a 0. Force Capture Event 1 0h = No effect. 1h = Writing a 1 sets the CEVT1 flag bit.
0	RESERVED	R	0h	

15.3.4.1.13 REVID Register (offset = 5Ch) [reset = 44D22100h]

REVID is shown in [Figure 15-174](#) and described in [Table 15-146](#).

Figure 15-129. REVID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-44D22100h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-119. REVID Register Field Descriptions

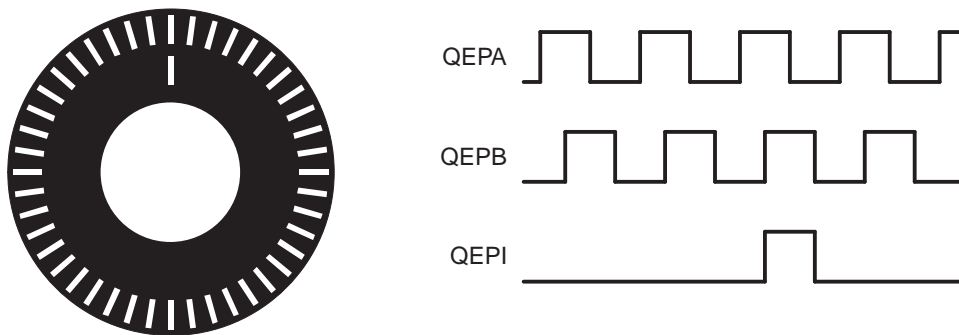
Bit	Field	Type	Reset	Description
31-0	REV	R	44D22100h	Revision ID.

15.4 Enhanced Quadrature Encoder Pulse (eQEP) Module

15.4.1 Introduction

A single track of slots patterns the periphery of an incremental encoder disk, as shown in [Figure 15-130](#). These slots create an alternating pattern of dark and light lines. The disk count is defined as the number of dark/light line pairs that occur per revolution (lines per revolution). As a rule, a second track is added to generate a signal that occurs once per revolution (index signal: QEPI), which can be used to indicate an absolute position. Encoder manufacturers identify the index pulse using different terms such as index, marker, home position, and zero reference.

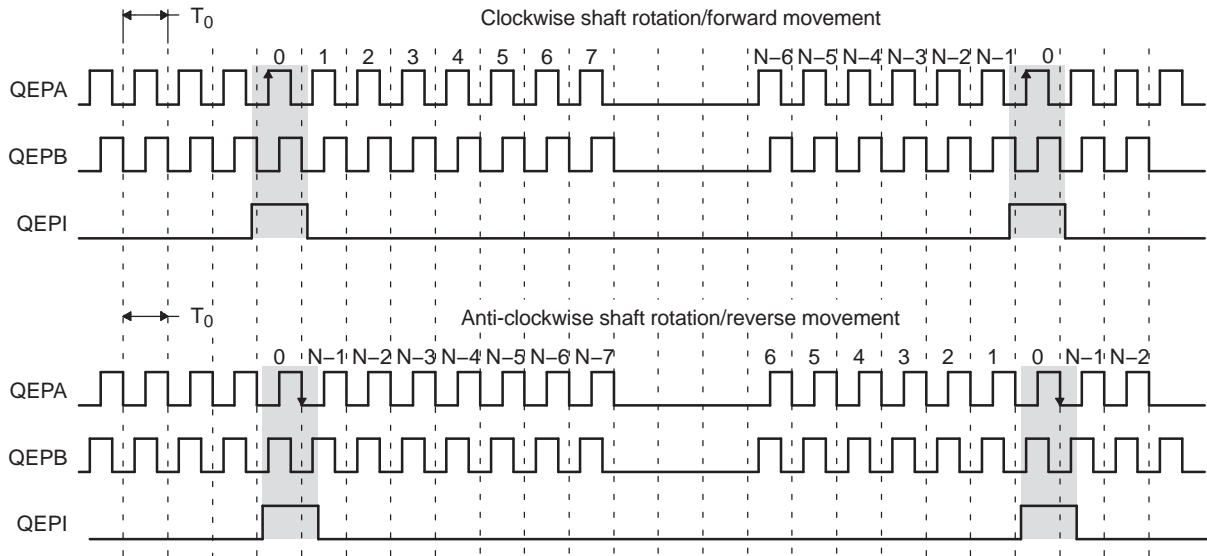
Figure 15-130. Optical Encoder Disk



To derive direction information, the lines on the disk are read out by two different photo-elements that "look" at the disk pattern with a mechanical shift of 1/4 the pitch of a line pair between them. This shift is realized with a reticle or mask that restricts the view of the photo-element to the desired part of the disk lines. As the disk rotates, the two photo-elements generate signals that are shifted 90 degrees out of phase from each other. These are commonly called the quadrature QEPA and QEPB signals. The clockwise direction for most encoders is defined as the QEPA channel going positive before the QEPB channel and vice versa as shown in [Figure 15-131](#).

The encoder wheel typically makes one revolution for every revolution of the motor or the wheel may be at a geared rotation ratio with respect to the motor. Therefore, the frequency of the digital signal coming from the QEPA and QEPB outputs varies proportionally with the velocity of the motor. For example, a 2000-line encoder directly coupled to a motor running at 5000 revolutions per minute (rpm) results in a frequency of 166.6 KHz, so by measuring the frequency of either the QEPA or QEPB output, the processor can determine the velocity of the motor.

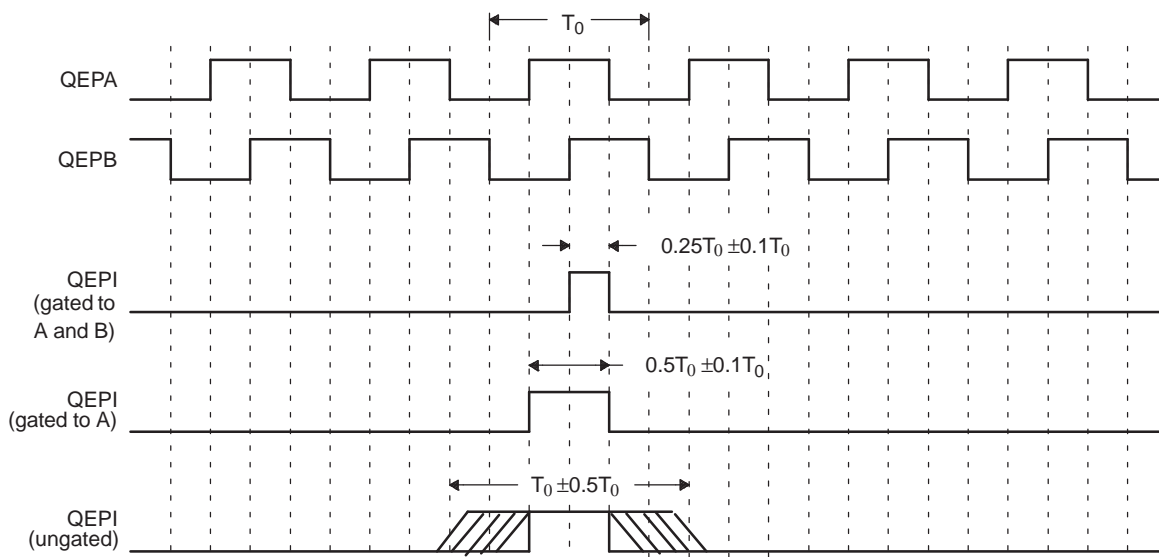
Figure 15-131. QEP Encoder Output Signal for Forward/Reverse Movement



Legend: N = lines per revolution

Quadrature encoders from different manufacturers come with two forms of index pulse (gated index pulse or ungated index pulse) as shown in Figure 15-132. A nonstandard form of index pulse is ungated. In the ungated configuration, the index edges are not necessarily coincident with A and B signals. The gated index pulse is aligned to any of the four quadrature edges and width of the index pulse and can be equal to a quarter, half, or full period of the quadrature signal.

Figure 15-132. Index Pulse Example



Some typical applications of shaft encoders include robotics and even computer input in the form of a mouse. Inside your mouse you can see where the mouse ball spins a pair of axles (a left/right, and an up/down axle). These axles are connected to optical shaft encoders that effectively tell the computer how fast and in what direction the mouse is moving.

General Issues: Estimating velocity from a digital position sensor is a cost-effective strategy in motor control. Two different first order approximations for velocity may be written as:

$$v(k) \approx \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \quad (1)$$

$$v(k) \approx \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T} \quad (2)$$

where

$v(k)$: Velocity at time instant k

$x(k)$: Position at time instant k

$x(k-1)$: Position at time instant $k - 1$

T : Fixed unit time or inverse of velocity calculation rate

ΔX : Incremental position movement in unit time

$t(k)$: Time instant " k "

$t(k-1)$: Time instant " $k - 1$ "

X : Fixed unit position

ΔT : Incremental time elapsed for unit position movement.

[Equation 1](#) is the conventional approach to velocity estimation and it requires a time base to provide unit time event for velocity calculation. Unit time is basically the inverse of the velocity calculation rate.

The encoder count (position) is read once during each unit time event. The quantity $[x(k) - x(k-1)]$ is formed by subtracting the previous reading from the current reading. Then the velocity estimate is computed by multiplying by the known constant $1/T$ (where T is the constant time between unit time events and is known in advance).

Estimation based on [Equation 1](#) has an inherent accuracy limit directly related to the resolution of the position sensor and the unit time period T . For example, consider a 500-line per revolution quadrature encoder with a velocity calculation rate of 400 Hz. When used for position the quadrature encoder gives a four-fold increase in resolution, in this case, 2000 counts per revolution. The minimum rotation that can be detected is therefore 0.0005 revolutions, which gives a velocity resolution of 12 rpm when sampled at 400 Hz. While this resolution may be satisfactory at moderate or high speeds, for example, 1% error at 1200 rpm, it would clearly prove inadequate at low speeds. In fact, at speeds below 12 rpm, the speed estimate would erroneously be zero much of the time.

At low speed, [Equation 2](#) provides a more accurate approach. It requires a position sensor that outputs a fixed interval pulse train, such as the aforementioned quadrature encoder. The width of each pulse is defined by motor speed for a given sensor resolution. [Equation 2](#) can be used to calculate motor speed by measuring the elapsed time between successive quadrature pulse edges. However, this method suffers from the opposite limitation, as does [Equation 1](#). A combination of relatively large motor speeds and high sensor resolution makes the time interval ΔT small, and thus more greatly influenced by the timer resolution. This can introduce considerable error into high-speed estimates.

For systems with a large speed range (that is, speed estimation is needed at both low and high speeds), one approach is to use [Equation 2](#) at low speed and have the DSP software switch over to [Equation 1](#) when the motor speed rises above some specified threshold.

15.4.2 Functional Description

This section provides the eQEP inputs and functional description.

NOTE: Multiple identical eQEP modules can be contained in a system. The number of modules is device-dependent and is based on target application needs. In this document, the letter x within a signal or module name is used to indicate a generic eQEP instance on a device.

15.4.2.1 EQEP Inputs

The eQEP inputs include two pins for quadrature-clock mode or direction-count mode, an index (or 0 marker), and a strobe input.

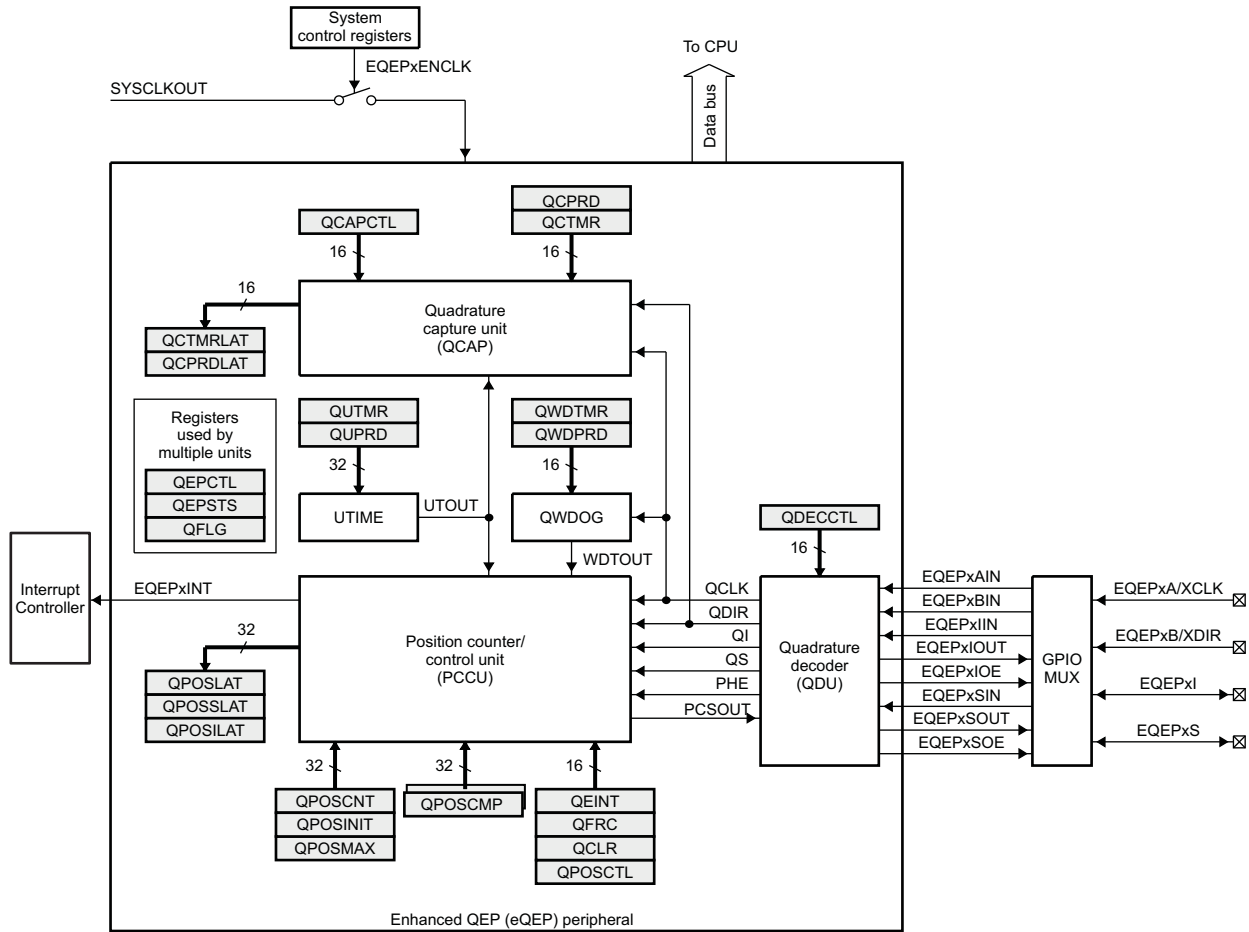
- **QEPA/XCLK and QEPB/XDIR:** These two pins can be used in quadrature-clock mode or direction-count mode.
 - **Quadrature-clock Mode:** The eQEP encoders provide two square wave signals (A and B) 90 electrical degrees out of phase whose phase relationship is used to determine the direction of rotation of the input shaft and number of eQEP pulses from the index position to derive the relative position information. For forward or clockwise rotation, QEPA signal leads QEPB signal and vice versa. The quadrature decoder uses these two inputs to generate quadrature-clock and direction signals.
 - **Direction-count Mode:** In direction-count mode, direction and clock signals are provided directly from the external source. Some position encoders have this type of output instead of quadrature output. The QEPA pin provides the clock input and the QEPB pin provides the direction input.
- **QEPI: Index or Zero Marker:** The eQEP encoder uses an index signal to assign an absolute start position from which position information is incrementally encoded using quadrature pulses. This pin is connected to the index output of the eQEP encoder to optionally reset the position counter for each revolution. This signal can be used to initialize or latch the position counter on the occurrence of a desired event on the index pin.
- **QEPS: Strobe Input:** This general-purpose strobe signal can initialize or latch the position counter on the occurrence of a desired event on the strobe pin. This signal is typically connected to a sensor or limit switch to notify that the motor has reached a defined position.

15.4.2.2 Functional Description

The eQEP peripheral contains the following major functional units (as shown in [Figure 15-133](#)):

- Programmable input qualification for each pin (part of the GPIO MUX)
- Quadrature decoder unit (QDU)
- Position counter and control unit for position measurement (PCCU)
- Quadrature edge-capture unit for low-speed measurement (QCAP)
- Unit time base for speed/frequency measurement (UTIME)
- Watchdog timer for detecting stalls (QWDOG)

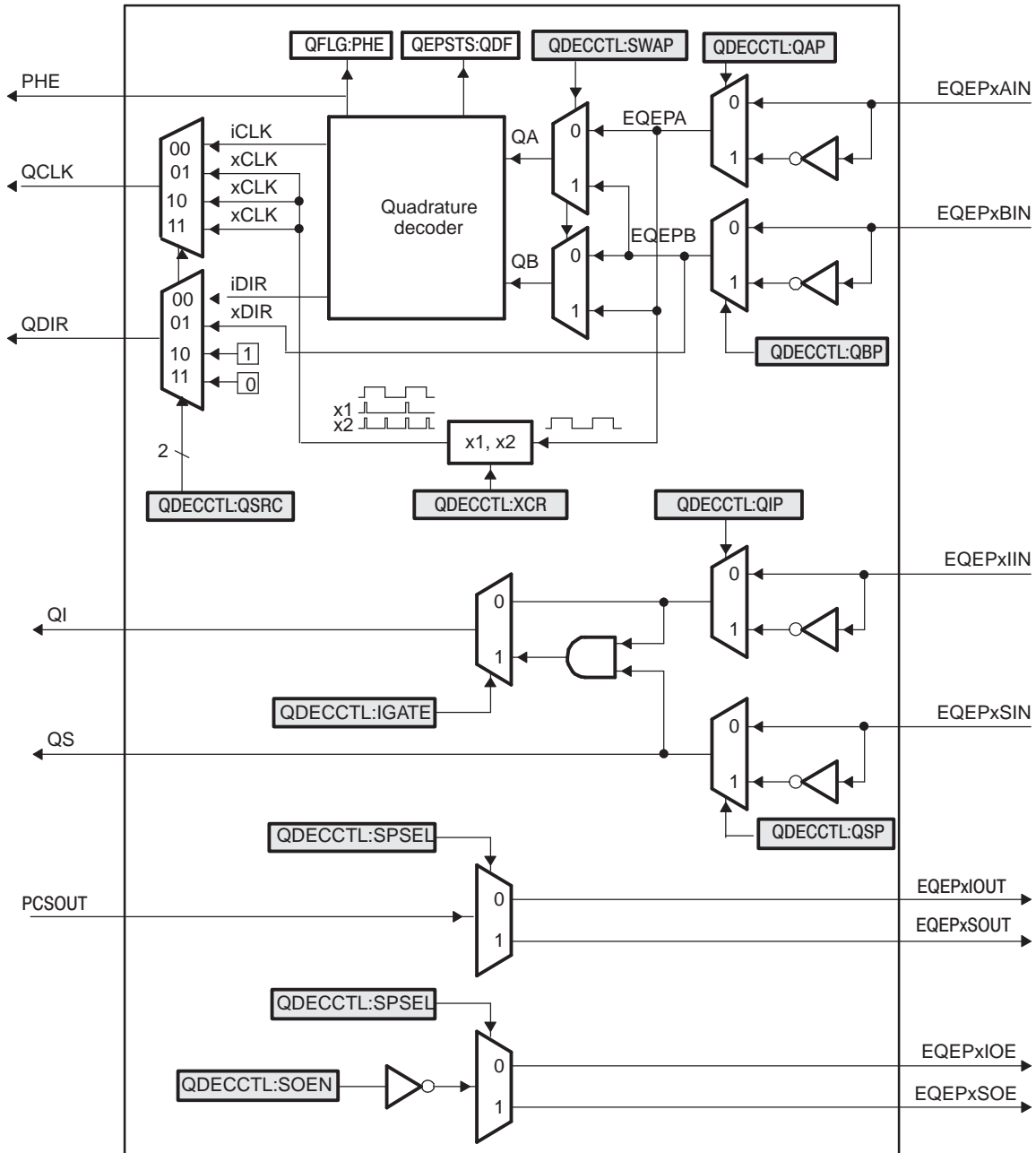
Figure 15-133. Functional Block Diagram of the eQEP Peripheral



15.4.2.3 Quadrature Decoder Unit (QDU)

Figure 15-134 shows a functional block diagram of the QDU.

Figure 15-134. Functional Block Diagram of Decoder Unit



15.4.2.3.1 Position Counter Input Modes

Clock and direction input to position counter is selected using the QSRC bit in the eQEP decoder control register (QDECCTL), based on interface input requirement as follows:

- Quadrature-count mode
- Direction-count mode
- UP-count mode
- DOWN-count mode

15.4.2.3.1.1 Quadrature Count Mode

The quadrature decoder generates the direction and clock to the position counter in quadrature count mode.

Direction Decoding— The direction decoding logic of the eQEP circuit determines which one of the sequences (QEPA, QEPB) is the leading sequence and accordingly updates the direction information in the QDF bit in the eQEP status register (QEPSTS). [Table 15-120](#) and [Figure 15-135](#) show the direction decoding logic in truth table and state machine form. Both edges of the QEPA and QEPB signals are sensed to generate count pulses for the position counter. Therefore, the frequency of the clock generated by the eQEP logic is four times that of each input sequence. [Figure 15-136](#) shows the direction decoding and clock generation from the eQEP input signals.

Phase Error Flag— In normal operating conditions, quadrature inputs QEPA and QEPB will be 90 degrees out of phase. The phase error flag (PHE) is set in the QFLG register when edge transition is detected simultaneously on the QEPA and QEPB signals to optionally generate interrupts. State transitions marked by dashed lines in [Figure 15-135](#) are invalid transitions that generate a phase error.

Count Multiplication— The eQEP position counter provides 4x times the resolution of an input clock by generating a quadrature-clock (QCLK) on the rising/falling edges of both eQEP input clocks (QEPA and QEPB) as shown in [Figure 15-136](#).

Reverse Count— In normal quadrature count operation, QEPA input is fed to the QA input of the quadrature decoder and the QEPB input is fed to the QB input of the quadrature decoder. Reverse counting is enabled by setting the SWAP bit in the eQEP decoder control register (QDECCTL). This will swap the input to the quadrature decoder thereby reversing the counting direction.

Table 15-120. Quadrature Decoder Truth Table

Previous Edge	Present Edge	QDIR	QPOSCNT
QA↑	QB↑	UP	Increment
	QB↓	DOWN	Decrement
	QA↓	TOGGLE	Increment or Decrement
QA↓	QB↓	UP	Increment
	QB↑	DOWN	Decrement
	QA↑	TOGGLE	Increment or Decrement
QB↑	QA↑	DOWN	Increment
	QA↓	UP	Decrement
	QB↓	TOGGLE	Increment or Decrement
QB↓	QA↓	DOWN	Increment
	QA↑	UP	Decrement
	QB↑	TOGGLE	Increment or Decrement

Figure 15-135. Quadrature Decoder State Machine

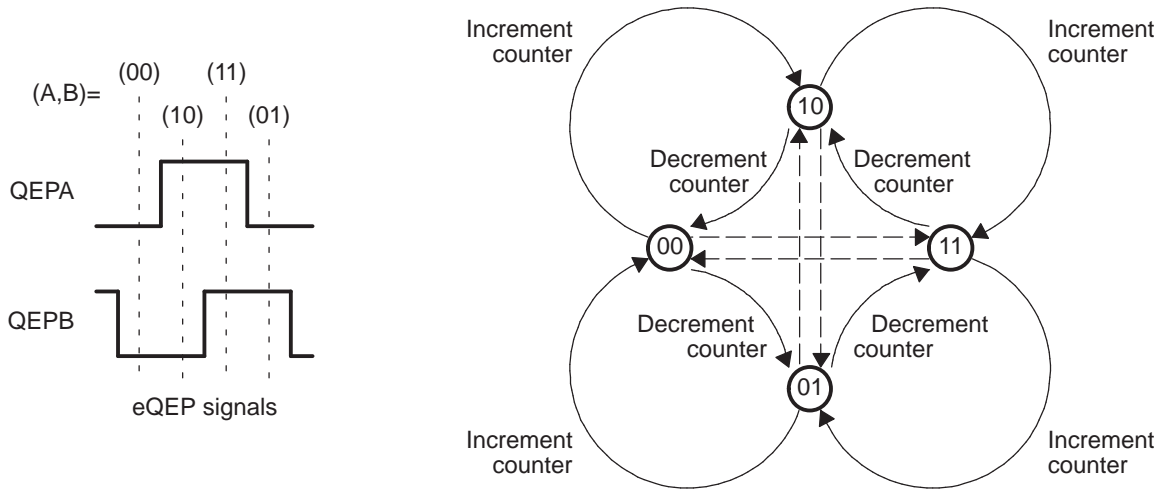
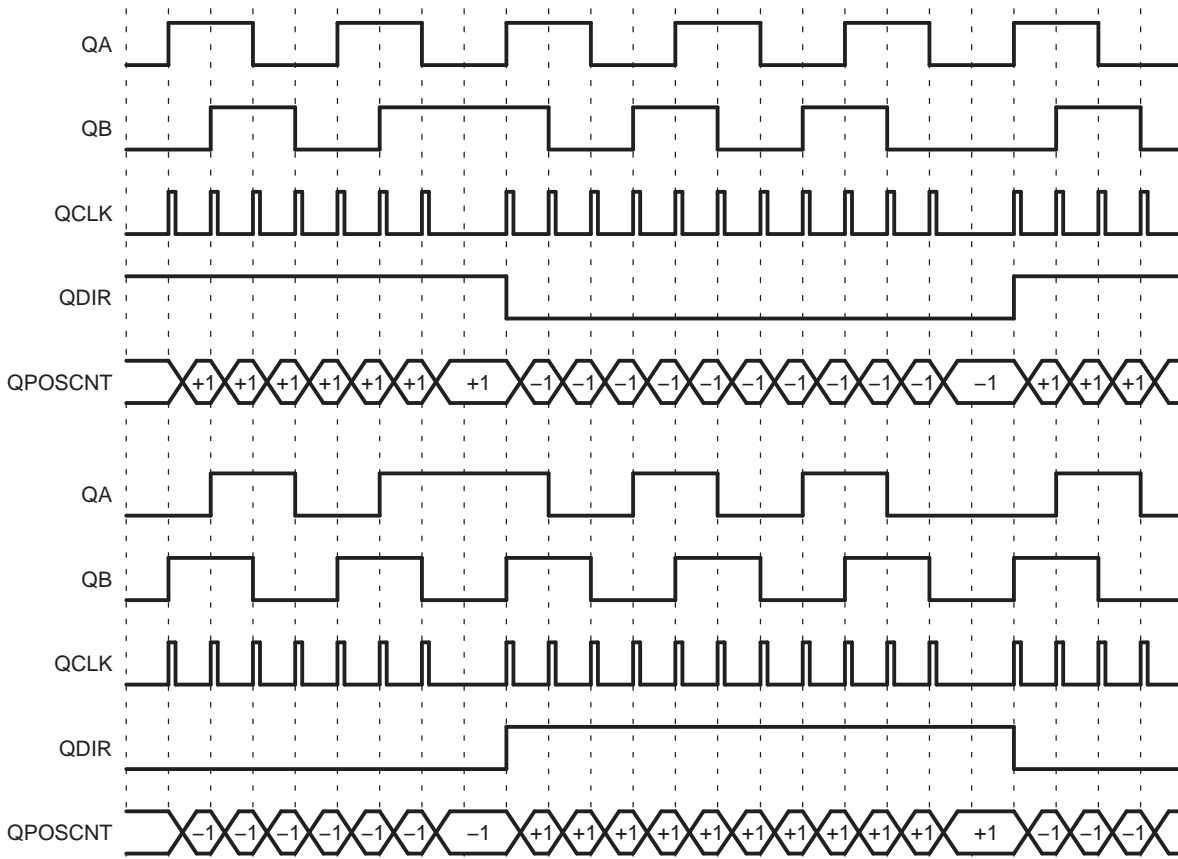


Figure 15-136. Quadrature-clock and Direction Decoding



15.4.2.3.1.2 Direction-count Mode

Some position encoders provide direction and clock outputs, instead of quadrature outputs. In such cases, direction-count mode can be used. QEPA input will provide the clock for position counter and the QEPB input will have the direction information. The position counter is incremented on every rising edge of a QEPA input when the direction input is high and decremented when the direction input is low.

15.4.2.3.1.3 Up-Count Mode

The counter direction signal is hard-wired for up count and the position counter is used to measure the frequency of the QEPA input. Setting of the XCR bit in the eQEP decoder control register (QDECCTL) enables clock generation to the position counter on both edges of the QEPA input, thereby increasing the measurement resolution by 2x factor.

15.4.2.3.1.4 Down-Count Mode

The counter direction signal is hardwired for a down count and the position counter is used to measure the frequency of the QEPA input. Setting of the XCR bit in the eQEP decoder control register (QDECCTL) enables clock generation to the position counter on both edges of a QEPA input, thereby increasing the measurement resolution by 2x factor.

15.4.2.3.2 eQEP Input Polarity Selection

Each eQEP input can be inverted using the in the eQEP decoder control register (QDECCTL[8:5]) control bits. As an example, setting of the QIP bit in QDECCTL inverts the index input.

15.4.2.3.3 Position-Compare Sync Output

The eQEP peripheral includes a position-compare unit that is used to generate the position-compare sync signal on compare match between the position counter register (QPOSCNT) and the position-compare register (QPOSCMP). This sync signal can be output using an index pin or strobe pin of the EQEP peripheral.

Setting the SOEN bit in the eQEP decoder control register (QDECCTL) enables the position-compare sync output and the SPSEL bit in QDECCTL selects either an eQEP index pin or an eQEP strobe pin.

15.4.2.4 Position Counter and Control Unit (PCCU)

The position counter and control unit provides two configuration registers (QEPCTL and QPOSCTL) for setting up position counter operational modes, position counter initialization/latch modes and position-compare logic for sync signal generation.

15.4.2.4.1 Position Counter Operating Modes

Position counter data may be captured in different manners. In some systems, the position counter is accumulated continuously for multiple revolutions and the position counter value provides the position information with respect to the known reference. An example of this is the quadrature encoder mounted on the motor controlling the print head in the printer. Here the position counter is reset by moving the print head to the home position and then position counter provides absolute position information with respect to home position.

In other systems, the position counter is reset on every revolution using index pulse and position counter provides rotor angle with respect to index pulse position.

Position counter can be configured to operate in following four modes

- Position Counter Reset on Index Event
- Position Counter Reset on Maximum Position
- Position Counter Reset on the first Index Event
- Position Counter Reset on Unit Time Out Event (Frequency Measurement)

In all the above operating modes, position counter is reset to 0 on overflow and to QPOSMAX register value on underflow. Overflow occurs when the position counter counts up after QPOSMAX value. Underflow occurs when position counter counts down after "0". Interrupt flag is set to indicate overflow/underflow in QFLG register.

15.4.2.4.1.1 Position Counter Reset on Index Event (QEPCTL[PCRM] = 00)

If the index event occurs during the forward movement, then position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the QPOSMAX register on the next eQEP clock.

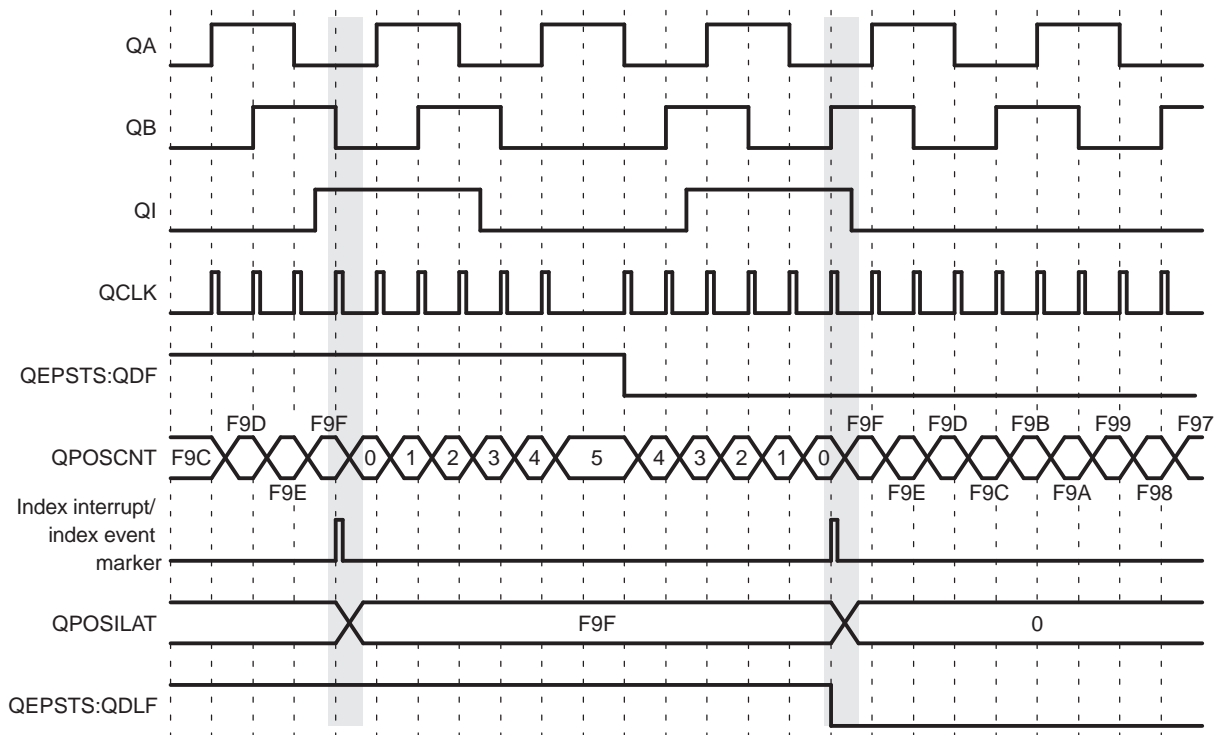
First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in QEPSTS registers, it also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for index event reset operation.

For example, if the first reset operation occurs on the falling edge of QEPB during the forward direction, then all the subsequent reset must be aligned with the falling edge of QEPB for the forward rotation and on the rising edge of QEPB for the reverse rotation as shown in [Figure 15-137](#).

The position-counter value is latched to the QPOSILAT register and direction information is recorded in the QEPSTS[QDLF] bit on every index event marker. The position-counter error flag (QEPSTS[PCEF]) and error interrupt flag (QFLG[PCE]) are set if the latched value is not equal to 0 or QPOSMAX. The position-counter error flag (QEPSTS[PCEF]) is updated on every index event marker and an interrupt flag (QFLG[PCE]) will be set on error that can be cleared only through software.

The index event latch configuration QEPCTL[IEL] bits are ignored in this mode and position counter error flag/interrupt flag are generated only in index event reset mode.

Figure 15-137. Position Counter Reset by Index Pulse for 1000 Line Encoder (QPOSMAX = 3999 or F9Fh)

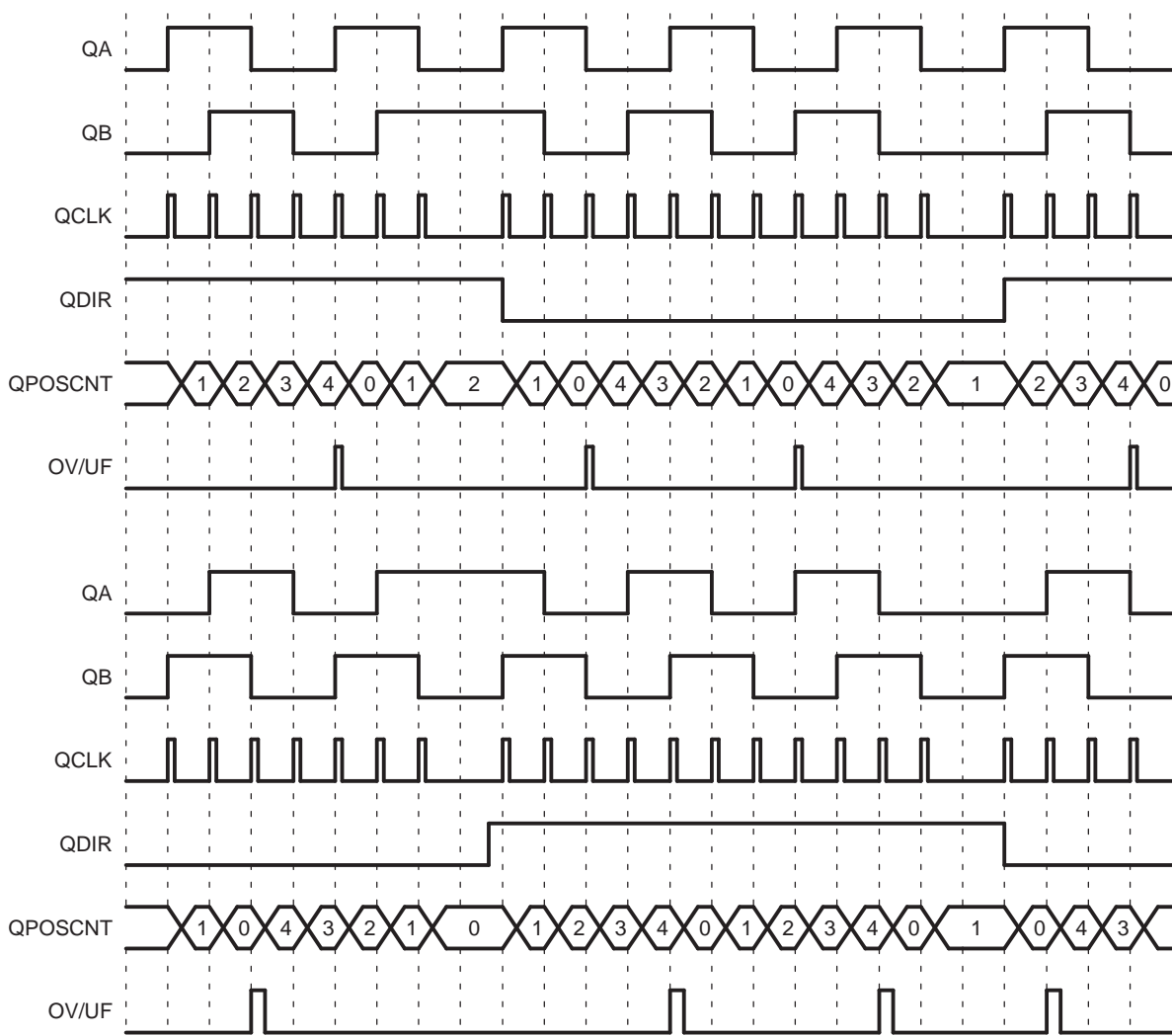


15.4.2.4.1.2 Position Counter Reset on Maximum Position (QEPCTL[PCRM]=01)

If the position counter is equal to QPOS MAX, then the position counter is reset to 0 on the next eQEP clock for forward movement and position counter overflow flag is set. If the position counter is equal to ZERO, then the position counter is reset to QPOS MAX on the next QEP clock for reverse movement and position counter underflow flag is set. Figure 15-138 shows the position counter reset operation in this mode.

First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in the QEPSTS registers; it also remembers the quadrature edge on the first index marker so that the same relative quadrature transition is used for the software index marker (QEPCTL[IEL]=11).

Figure 15-138. Position Counter Underflow/Overflow (QPOS MAX = 4)



15.4.2.4.1.3 Position Counter Reset on the First Index Event (QEPCTL[PCRM] = 10)

If the index event occurs during forward movement, then the position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the QPOSMAX register on the next eQEP clock. Note that this is done only on the first occurrence and subsequently the position counter value is not reset on an index event; rather, it is reset based on maximum position as described in [Section 15.4.2.4.1.2](#).

First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in QEPSTS registers. It also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for software index marker (QEPCTL[IEL]=11).

15.4.2.4.1.4 Position Counter Reset on Unit Time out Event (QEPCTL[PCRM] = 11)

In this mode, the QPOSCNT value is latched to the QPOSLAT register and then the QPOSCNT is reset (to 0 or QPOSMAX, depending on the direction mode selected by QDECCTL[QSRC] bits on a unit time event). This is useful for frequency measurement.

15.4.2.4.2 Position Counter Latch

The eQEP index and strobe input can be configured to latch the position counter (QPOSCNT) into QPOSILAT and QPOSSLAT, respectively, on occurrence of a definite event on these pins.

15.4.2.4.2.1 Index Event Latch

In some applications, it may not be desirable to reset the position counter on every index event and instead it may be required to operate the position counter in full 32-bit mode (QEPCTL[PCRM] = 01 and QEPCTL[PCRM] = 10 modes).

In such cases, the eQEP position counter can be configured to latch on the following events and direction information is recorded in the QEPSTS[QDLF] bit on every index event marker.

- Latch on Rising edge (QEPCTL[IEL] = 01)
- Latch on Falling edge (QEPCTL[IEL] = 10)
- Latch on Index Event Marker (QEPCTL[IEL] = 11)

This is particularly useful as an error checking mechanism to check if the position counter accumulated the correct number of counts between index events. As an example, the 1000-line encoder must count 4000 times when moving in the same direction between the index events.

The index event latch interrupt flag (QFLG[IEL]) is set when the position counter is latched to the QPOSILAT register. The index event latch configuration bits (QEPCTZ[IEL]) are ignored when QEPCTL[PCRM] = 00.

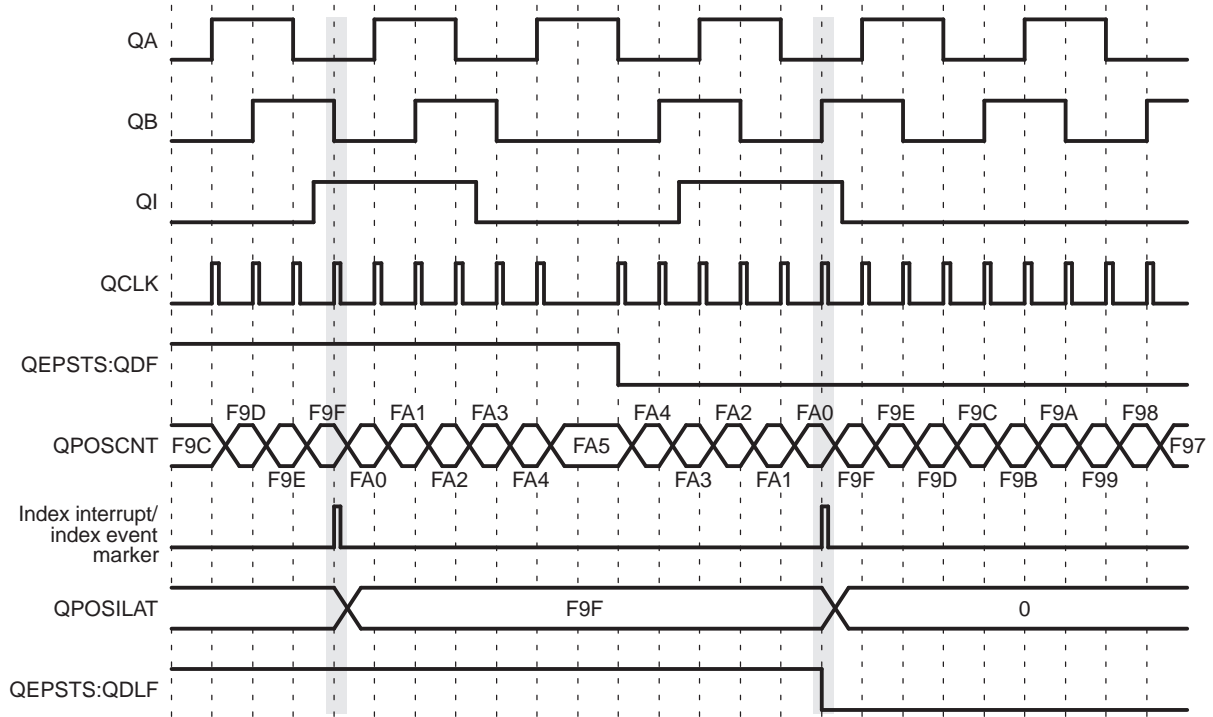
Latch on Rising Edge (QEPCTL[IEL] = 01)— The position counter value (QPOSCNT) is latched to the QPOSILAT register on every rising edge of an index input.

Latch on Falling Edge (QEPCTL[IEL] = 10)— The position counter value (QPOSCNT) is latched to the QPOSILAT register on every falling edge of index input.

Latch on Index Event Marker/Software Index Marker (QEPCTL[IEL] = 11)— The first index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in the QEPSTS registers. It also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for latching the position counter (QEPCTL[IEL] = 11).

[Figure 15-139](#) shows the position counter latch using an index event marker.

Figure 15-139. Software Index Marker for 1000-line Encoder (QEPCTL[IEL] = 1)



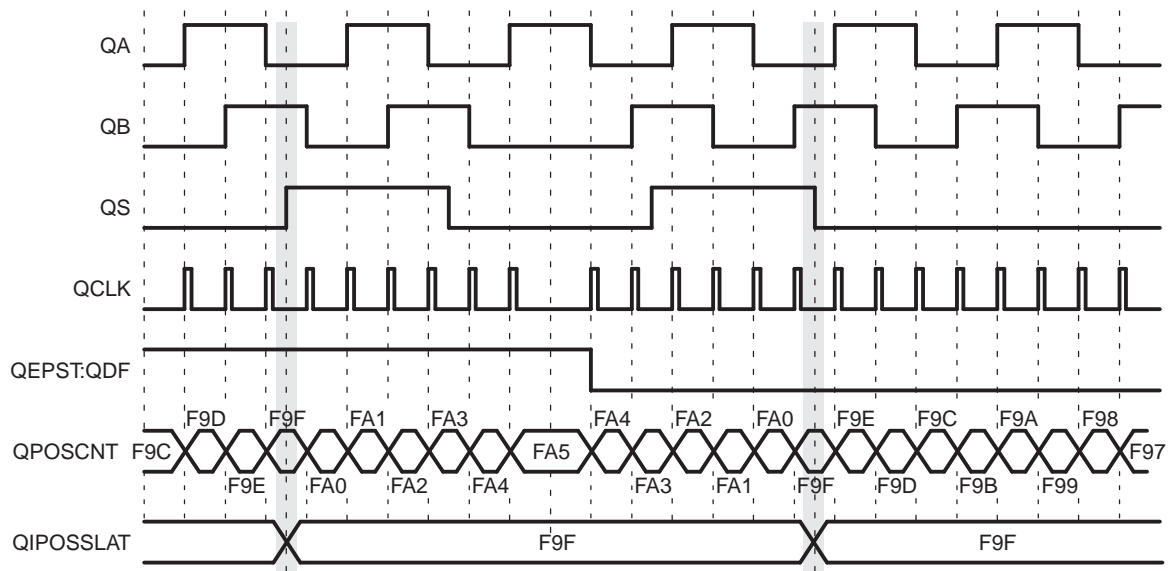
15.4.2.4.2.2 Strobe Event Latch

The position-counter value is latched to the QPOSSLAT register on the rising edge of the strobe input by clearing the QEPCTL[SEL] bit.

If the QEPCTL[SEL] bit is set, then the position counter value is latched to the QPOSSLAT register on the rising edge of the strobe input for forward direction and on the falling edge of the strobe input for reverse direction as shown in Figure 15-140.

The strobe event latch interrupt flag (QFLG[SEL]) is set when the position counter is latched to the QPOSSLAT register.

Figure 15-140. Strobe Event Latch (QEPCTL[SEL] = 1)



15.4.2.4.3 Position Counter Initialization

The position counter can be initialized using following events:

- Index event
- Strobe event
- Software initialization

Index Event Initialization (IEI)— The QEPI index input can be used to trigger the initialization of the position counter at the rising or falling edge of the index input.

If the QEPCTL[IEI] bits are 10, then the position counter (QPOSCNT) is initialized with a value in the QPOSINIT register on the rising edge of strobe input for forward direction and on the falling edge of strobe input for reverse direction.

The index event initialization interrupt flag (QFLG[IEI]) is set when the position counter is initialized with a value in the QPOSINIT register.

Strobe Event Initialization (SEI)— If the QEPCTL[SEI] bits are 10, then the position counter is initialized with a value in the QPOSINIT register on the rising edge of strobe input.

If the QEPCTL[SEL] bits are 11, then the position counter (QPOSCNT) is initialized with a value in the QPOSINIT register on the rising edge of strobe input for forward direction and on the falling edge of strobe input for reverse direction.

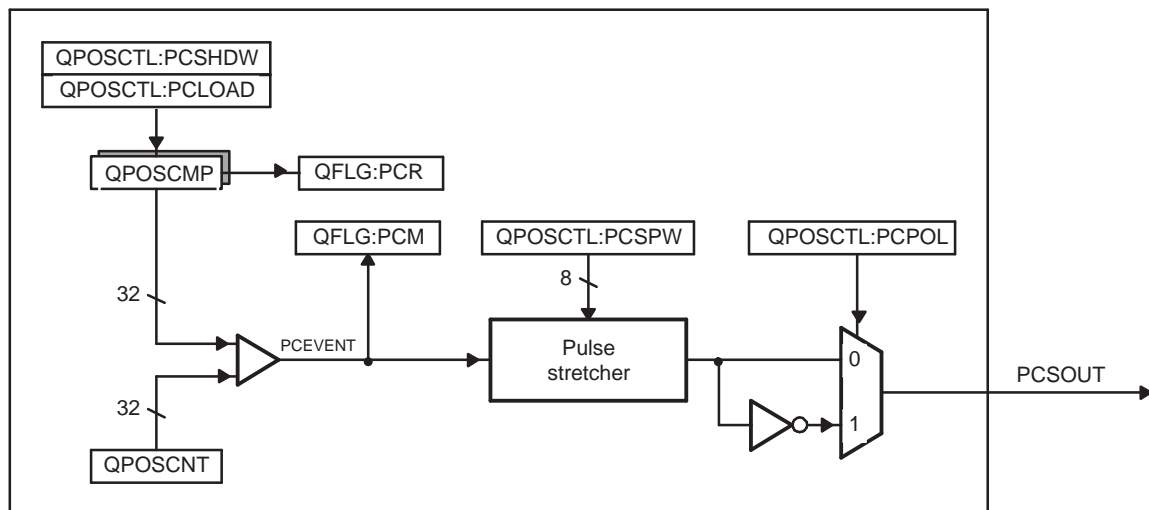
The strobe event initialization interrupt flag (QFLG[SEI]) is set when the position counter is initialized with a value in the QPOSINIT register.

Software Initialization (SWI)— The position counter can be initialized in software by writing a 1 to the QEPCTL[SWI] bit, which will automatically be cleared after initialization.

15.4.2.4.4 eQEP Position-compare Unit

The eQEP peripheral includes a position-compare unit that is used to generate a sync output and/or interrupt on a position-compare match. Figure 15-141 shows a diagram. The position-compare (QPOSCMP) register is shadowed and shadow mode can be enabled or disabled using the QPOSCTL[PSSHDW] bit. If the shadow mode is not enabled, the CPU writes directly to the active position compare register.

Figure 15-141. eQEP Position-compare Unit



In shadow mode, you can configure the position-compare unit (QPOSCTL[PCLOAD]) to load the shadow register value into the active register on the following events and to generate the position-compare ready (QFLG[PCR]) interrupt after loading.

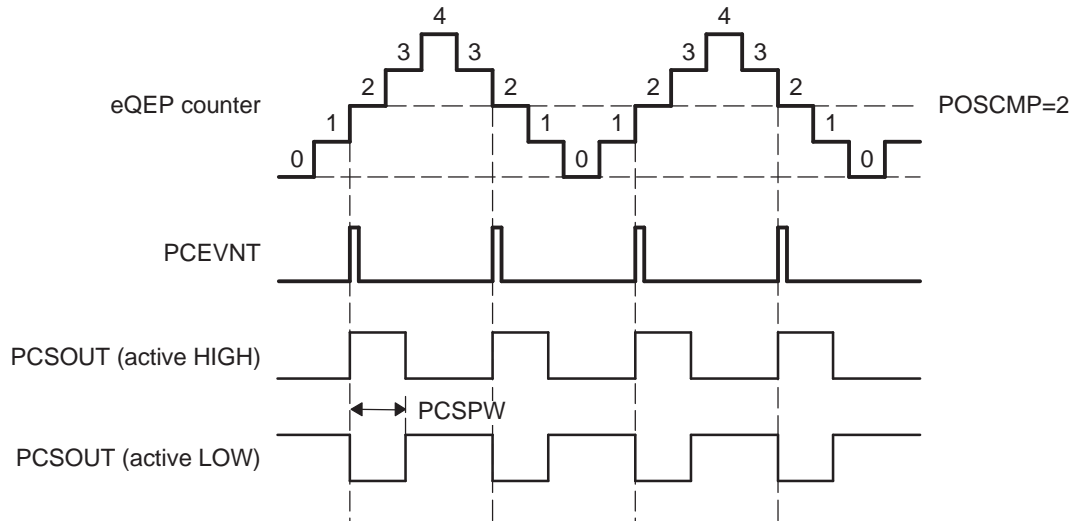
- Load on compare match
- Load on position-counter zero event

The position-compare match (QFLG[PCM]) is set when the position-counter value (QPOSCNT) matches with the active position-compare register (QPOSCMP) and the position-compare sync output of the programmable pulse width is generated on compare match to trigger an external device.

For example, if QPOSCMP = 2, the position-compare unit generates a position-compare event on 1 to 2 transitions of the eQEP position counter for forward counting direction and on 3 to 2 transitions of the eQEP position counter for reverse counting direction (see Figure 15-142).

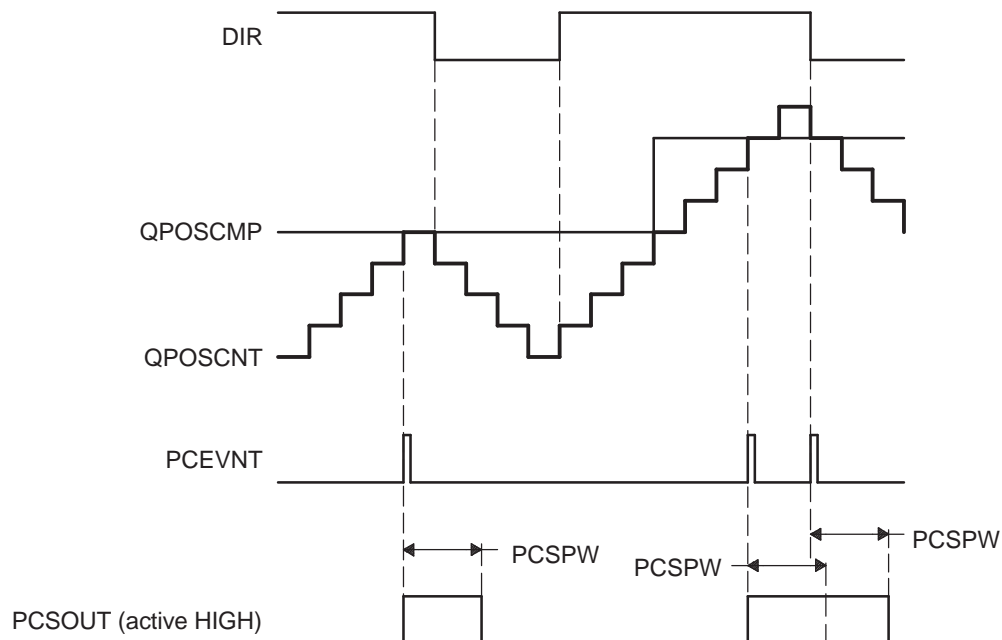
shows the layout of the eQEP Position-Compare Control Register (QPOSCTL) and describes the QPOSCTL bit fields.

Figure 15-142. eQEP Position-compare Event Generation Points



The pulse stretcher logic in the position-compare unit generates a programmable position-compare sync pulse output on the position-compare match. In the event of a new position-compare match while a previous position-compare pulse is still active, then the pulse stretcher generates a pulse of specified duration from the new position-compare event as shown in Figure 15-143.

Figure 15-143. eQEP Position-compare Sync Output Pulse Stretcher



15.4.2.5 eQEP Edge Capture Unit

The eQEP peripheral includes an integrated edge capture unit to measure the elapsed time between the unit position events as shown in [Figure 15-144](#). This feature is typically used for low speed measurement using the following equation:

$$v(k) = \frac{X}{t(k) - t(k - 1)} = \frac{X}{\Delta T} \quad (3)$$

where,

- X - Unit position is defined by integer multiple of quadrature edges (see [Figure 15-145](#))
- ΔT - Elapsed time between unit position events
- v(k) - Velocity at time instant "k"

The eQEP capture timer (QCTMR) runs from prescaled SYSCLKOUT and the prescaler is programmed by the QCAPCTL[CCPS] bits. The capture timer (QCTMR) value is latched into the capture period register (QCPRD) on every unit position event and then the capture timer is reset, a flag is set in QEPSTS[UPEVNT] to indicate that new value is latched into the QCPRD register. Software can check this status flag before reading the period register for low speed measurement and clear the flag by writing 1.

Time measurement (ΔT) between unit position events will be correct if the following conditions are met:

- No more than 65,535 counts have occurred between unit position events.
- No direction change between unit position events.

The capture unit sets the eQEP overflow error flag (QEPSTS[COEF]) in the event of capture timer overflow between unit position events. If a direction change occurs between the unit position events, then an error flag is set in the status register (QEPSTS[CDEF]).

Capture Timer (QCTMR) and Capture period register (QCPRD) can be configured to latch on following events.

- CPU read of QPOSCNT register
- Unit time-out event

If the QEPCTL[QCLM] bit is cleared, then the capture timer and capture period values are latched into the QCTMRLAT and QCPRDLAT registers, respectively, when the CPU reads the position counter (QPOSCNT).

If the QEPCTL[QCLM] bit is set, then the position counter, capture timer, and capture period values are latched into the QPOSLAT, QCTMRLAT and QCPRDLAT registers, respectively, on unit time out.

[Figure 15-146](#) shows the capture unit operation along with the position counter.

NOTE: The QCAPCTL register should not be modified dynamically (such as switching CAPCLK prescaling mode from QCLK/4 to QCLK/8). The capture unit must be disabled before changing the prescaler.

Figure 15-144. eQEP Edge Capture Unit

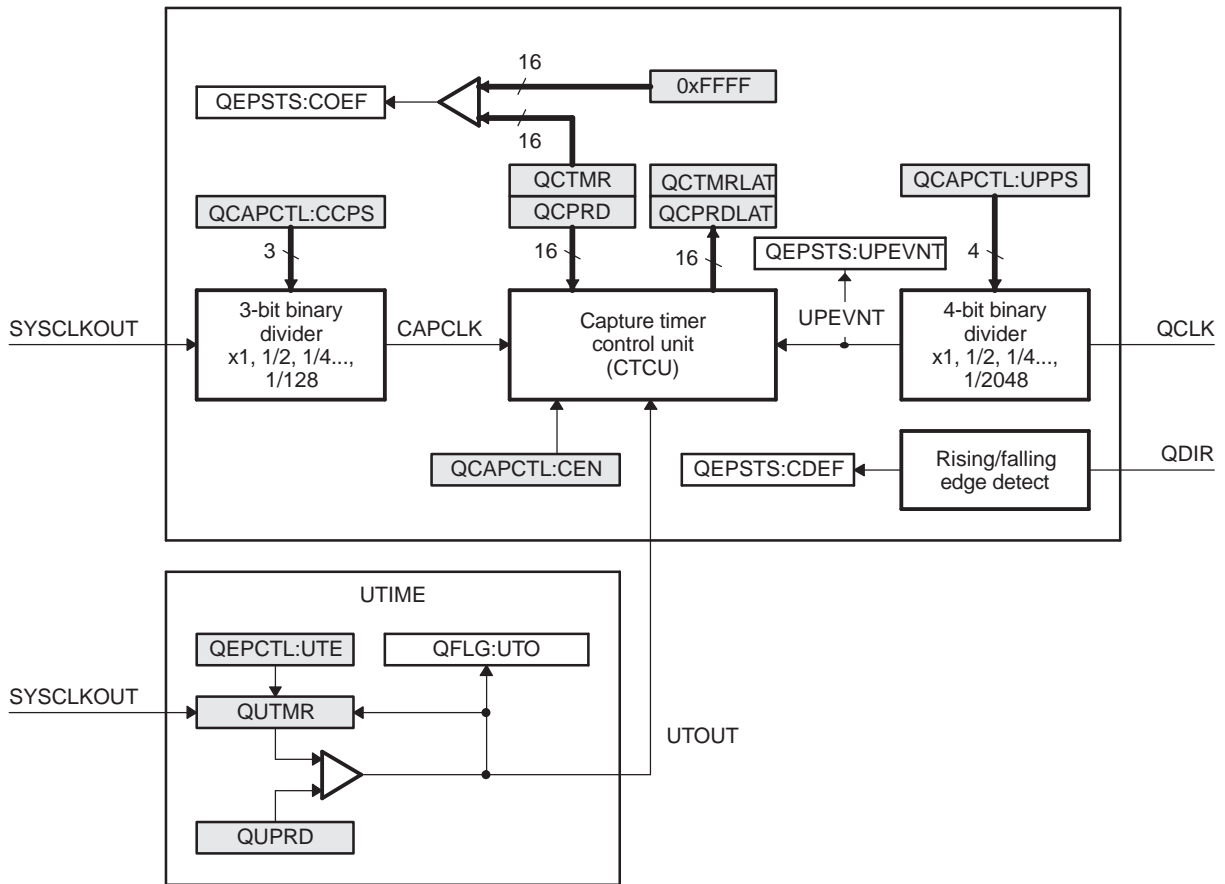
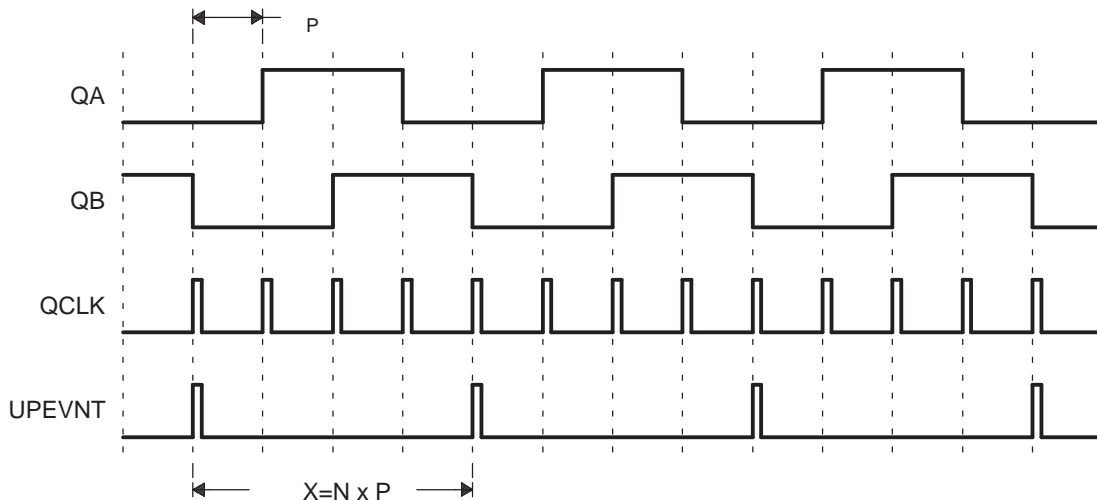
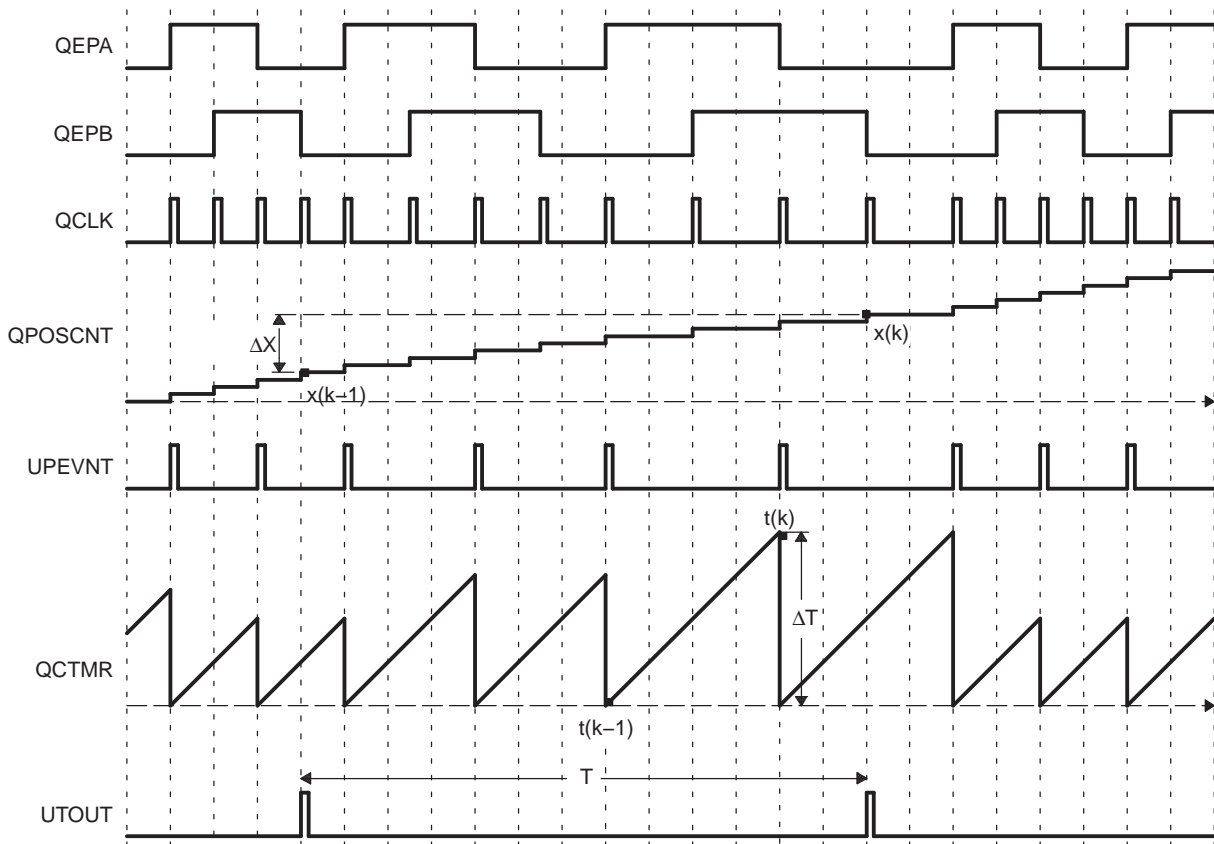


Figure 15-145. Unit Position Event for Low Speed Measurement (QCAPCTL[UPPS] = 0010)



N - Number of quadrature periods selected using QCAPCTL[UPPS] bits

Figure 15-146. eQEP Edge Capture Unit - Timing Details



Velocity Calculation Equations:

$$v(k) = \frac{x(k) - x(k - 1)}{T} = \frac{\Delta X}{T} \quad (4)$$

where

$v(k)$: Velocity at time instant k

$x(k)$: Position at time instant k

$x(k-1)$: Position at time instant $k - 1$

T : Fixed unit time or inverse of velocity calculation rate

ΔX : Incremental position movement in unit time

X : Fixed unit position

ΔT : Incremental time elapsed for unit position movement

$t(k)$: Time instant "k"

$t(k-1)$: Time instant "k - 1"

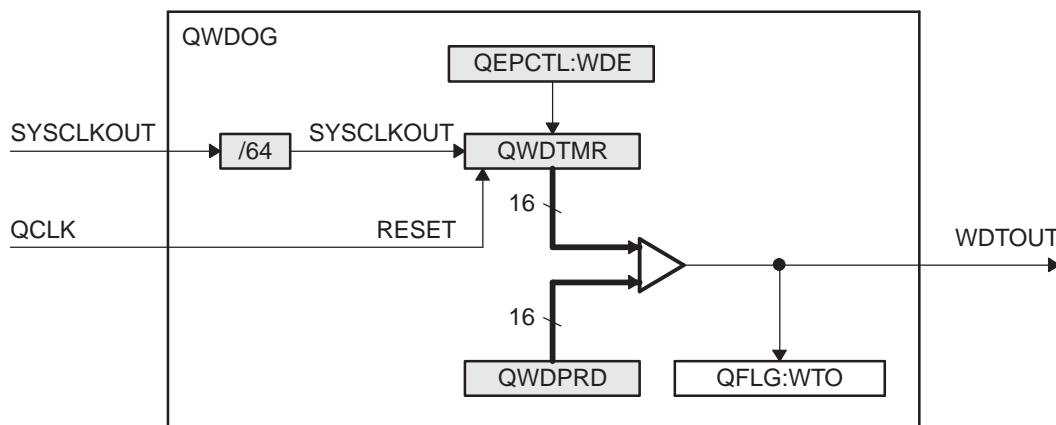
Unit time (T) and unit period (X) are configured using the QUPRD and QCAPCTL[UPPS] registers. Incremental position output and incremental time output is available in the QPOSLAT and QCPRDLAT registers.

Parameter	Relevant Register to Configure or Read the Information
T	Unit Period Register (QUPRD)
ΔX	Incremental Position = QOSLAT(k) - QOSLAT(K - 1)
X	Fixed unit position defined by sensor resolution and ZCAPCTL[UPPS] bits
ΔT	Capture Period Latch (QCPRDLAT)

15.4.2.6 eQEP Watchdog

The eQEP peripheral contains a 16-bit watchdog timer that monitors the quadrature-clock to indicate proper operation of the motion-control system. The eQEP watchdog timer is clocked from SYSCLKOUT/64 and the quadrature clock event (pulse) resets the watchdog timer. If no quadrature-clock event is detected until a period match ($QWDPRD = QWDTMR$), then the watchdog timer will time out and the watchdog interrupt flag will be set (QFLG[WTO]). The time-out value is programmable through the watchdog period register (QWDPRD).

Figure 15-147. eQEP Watchdog Timer

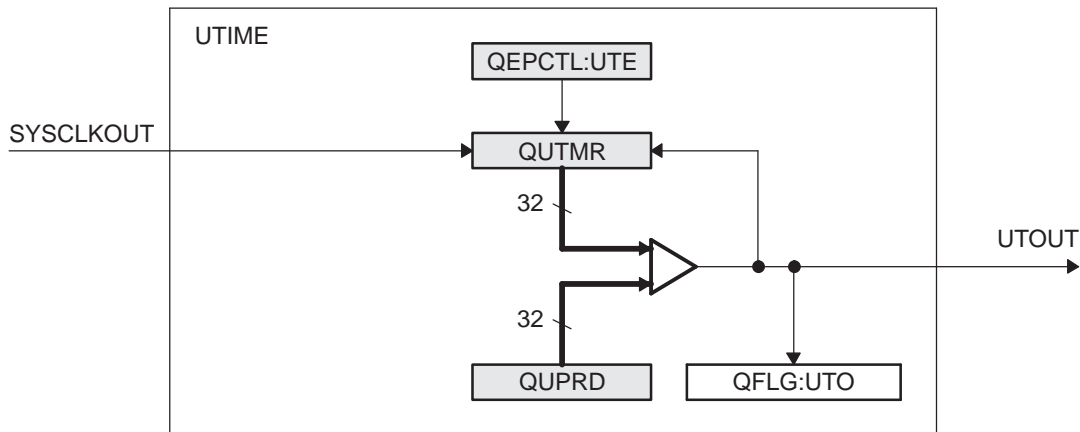


15.4.2.7 Unit Timer Base

The eQEP peripheral includes a 32-bit timer (QUTMR) that is clocked by SYSCLKOUT to generate periodic interrupts for velocity calculations. The unit time out interrupt is set (QFLG[UTO]) when the unit timer (QUTMR) matches the unit period register (QUPRD).

The eQEP peripheral can be configured to latch the position counter, capture timer, and capture period values on a unit time out event so that latched values are used for velocity calculation as described in Section [Section 15.4.2.5](#).

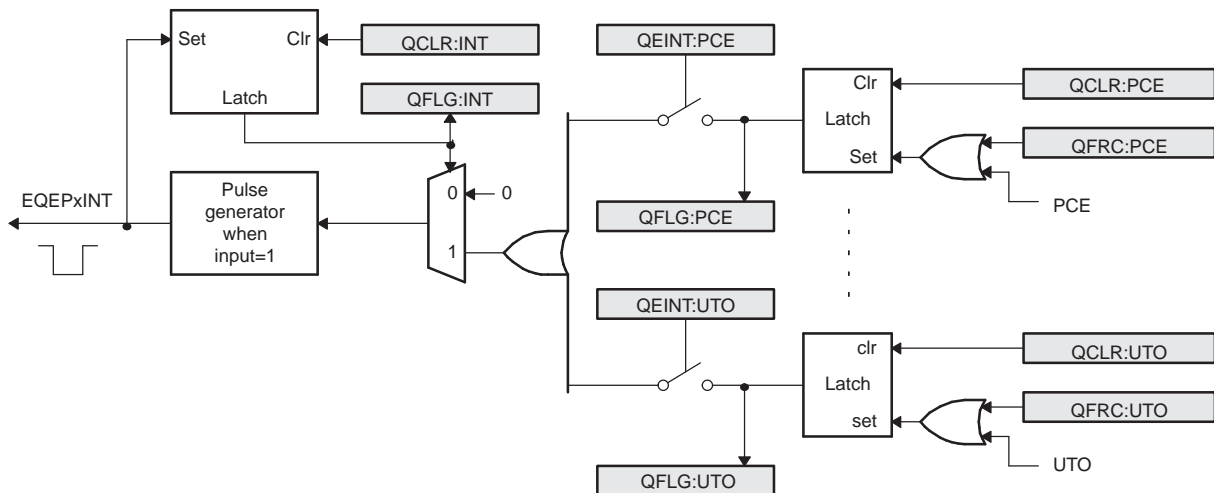
Figure 15-148. eQEP Unit Time Base



15.4.2.8 eQEP Interrupt Structure

Figure 15-149 shows how the interrupt mechanism works in the EQEP module.

Figure 15-149. EQEP Interrupt Generation



Eleven interrupt events (PCE, PHE, QDC, WTO, PCU, PCO, PCR, PCM, SEL, IEL, and UTO) can be generated. The interrupt control register (QEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (QFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit (INT). An interrupt pulse is generated only to the interrupt controller if any of the interrupt events is enabled, the flag bit is 1 and the INT flag bit is 0. The interrupt service routine will need to clear the global interrupt flag bit and the serviced event, via the interrupt clear register (QCLR), before any other interrupt pulses are generated. You can force an interrupt event by way of the interrupt force register (QFRC), which is useful for test purposes.

Note that the interrupts coming from the eQEP module are also used as DMA events. The interrupt registers should be used to enable and clear the current DMA event in order for the eQEP module to generate subsequent DMA events.

15.4.3 EQEP Registers

[Table 15-121](#) lists the memory-mapped registers for the EQEP. All register offset addresses not listed in [Table 15-121](#) should be considered as reserved locations and the register contents should not be modified.

Table 15-121. EQEP Registers

Offset	Acronym	Register Name	Section
0h	QPOSCNT	eQEP Position Counter Register	Section 15.4.3.1
4h	QPOSINIT	eQEP Position Counter Initialization Register	Section 15.4.3.2
8h	QPOSMAX	eQEP Maximum Position Count Register	Section 15.4.3.3
Ch	QPOSCMP	eQEP Position-Compare Register	Section 15.4.3.4
10h	QPOSILAT	eQEP Index Position Latch Register	Section 15.4.3.5
14h	QPOSSLAT	eQEP Strobe Position Latch Register	Section 15.4.3.6
18h	QPOSLAT	eQEP Position Counter Latch Register	Section 15.4.3.7
1Ch	QUTMR	eQEP Unit Timer Register	Section 15.4.3.8
20h	QUPRD	eQEP Unit Period Register	Section 15.4.3.9
24h	QWDTMR	eQEP Watchdog Timer Register	Section 15.4.3.10
26h	QWDPRD	eQEP Watchdog Period Register	Section 15.4.3.11
28h	QDECCTL	eQEP Decoder Control Register	Section 15.4.3.12
2Ah	QEPCTL	eQEP Control Register	Section 15.4.3.13
2Ch	QCAPCTL	eQEP Capture Control Register	Section 15.4.3.14
2Eh	QPOSCTL	eQEP Position-Compare Control Register	Section 15.4.3.15
30h	QEINT	eQEP Interrupt Enable Register	Section 15.4.3.16
32h	QFLG	eQEP Interrupt Flag Register	Section 15.4.3.17
34h	QCLR	eQEP Interrupt Clear Register	Section 15.4.3.18
36h	QFRC	eQEP Interrupt Force Register	Section 15.4.3.19
38h	QEPSTS	eQEP Status Register	Section 15.4.3.20
3Ah	QCTMR	eQEP Capture Timer Register	Section 15.4.3.21
3Ch	QCPRD	eQEP Capture Period Register	Section 15.4.3.22
3Eh	QCTMRLAT	eQEP Capture Timer Latch Register	Section 15.4.3.23
40h	QCPRDLAT	eQEP Capture Period Latch Register	Section 15.4.3.24
5Ch	REVID	eQEP Revision ID Register	Section 15.4.3.25

15.4.3.1 QPOSCNT Register (offset = 0h) [reset = 0h]

QPOSCNT is shown in [Figure 15-150](#) and described in [Table 15-122](#).

Figure 15-150. QPOSCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSCNT																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-122. QPOSCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSCNT	R/W	0h	This 32 bit position counter register counts up/down on every eQEP pulse based on direction input. This counter acts as a position integrator whose count value is proportional to position from a give reference point.

15.4.3.2 QPOSINIT Register (offset = 4h) [reset = 0h]

QPOSINIT is shown in [Figure 15-151](#) and described in [Table 15-123](#).

Figure 15-151. QPOSINIT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSINIT																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-123. QPOSINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSINIT	R/W	0h	This register contains the position value that is used to initialize the position counter based on external strobe or index event. The position counter can be initialized through software.

15.4.3.3 QPOSMAX Register (offset = 8h) [reset = 0h]

QPOSMAX is shown in [Figure 15-152](#) and described in [Table 15-124](#).

Figure 15-152. QPOSMAX Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSMAX																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-124. QPOSMAX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSMAX	R/W	0h	This register contains the maximum position counter value.

15.4.3.4 QPOSCMP Register (offset = Ch) [reset = 0h]

QPOSCMP is shown in [Figure 15-153](#) and described in [Table 15-125](#).

Figure 15-153. QPOSCMP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSCMP																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-125. QPOSCMP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSCMP	R/W	0h	The position-compare value in this register is compared with the position counter (QPOSCNT) to generate sync output and/or interrupt on compare match.

15.4.3.5 QPOSILAT Register (offset = 10h) [reset = 0h]

QPOSILAT is shown in [Figure 15-154](#) and described in [Table 15-126](#).

Figure 15-154. QPOSILAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSILAT																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-126. QPOSILAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSILAT	R	0h	The position-counter value is latched into this register on an index event as defined by the QEPCTL[IEL] bits.

15.4.3.6 QPOSSLAT Register (offset = 14h) [reset = 0h]

QPOSSLAT is shown in [Figure 15-155](#) and described in [Table 15-127](#).

Figure 15-155. QPOSSLAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSSLAT																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-127. QPOSSLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSSLAT	R	0h	The position-counter value is latched into this register on strobe event as defined by the QEPCTL[SEL] bits.

15.4.3.7 QPOSLAT Register (offset = 18h) [reset = 0h]

QPOSLAT is shown in [Figure 15-156](#) and described in [Table 15-128](#).

Figure 15-156. QPOSLAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSLAT																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-128. QPOSLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSLAT	R	0h	The position-counter value is latched into this register on unit time out event.

15.4.3.8 QUTMR Register (offset = 1Ch) [reset = 0h]

QUTMR is shown in [Figure 15-157](#) and described in [Table 15-129](#).

Figure 15-157. QUTMR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUTMR																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-129. QUTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QUTMR	R/W	0h	This register acts as time base for unit time event generation. When this timer value matches with unit time period value, unit time event is generated.

15.4.3.9 QUPRD Register (offset = 20h) [reset = 0h]

QUPRD is shown in [Figure 15-158](#) and described in [Table 15-130](#).

Figure 15-158. QUPRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUPRD																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-130. QUPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QUPRD	R/W	0h	This register contains the period count for unit timer to generate periodic unit time events to latch the eQEP position information at periodic interval and optionally to generate interrupt.

15.4.3.10 QWDTMR Register (offset = 24h) [reset = 0h]

QWDTMR is shown in [Figure 15-159](#) and described in [Table 15-131](#).

Figure 15-159. QWDTMR Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QWDTMR															
R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-131. QWDTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QWDTMR	R/W	0h	This register acts as time base for watch dog to detect motor stalls. When this timer value matches with watch dog period value, watch dog timeout interrupt is generated. This register is reset upon edge transition in quadrature-clock indicating the motion.

15.4.3.11 QWDPRD Register (offset = 26h) [reset = 0h]

QWDPRD is shown in [Figure 15-160](#) and described in [Table 15-132](#).

Figure 15-160. QWDPRD Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QWDPRD															
R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-132. QWDPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QWDPRD	R/W	0h	This register contains the time-out count for the eQEP peripheral watch dog timer. When the watchdog timer value matches the watchdog period value, a watchdog timeout interrupt is generated.

15.4.3.12 QDECCTL Register (offset = 28h) [reset = 0h]

 QDECCTL is shown in [Figure 15-161](#) and described in [Table 15-133](#).

Figure 15-161. QDECCTL Register

15		14		13		12		11		10		9		8	
QSRC				SOEN		SPSEL		XCR		SWAP		IGATE		QAP	
R/W-0h				R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7		6		5		4		3		2		1		0	
QBP		QIP		QSP		RESERVED									
R/W-0h		R/W-0h		R/W-0h		R-0h									

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-133. QDECCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	QSRC	R/W	0h	Position-counter source selection. 0h = Quadrature count mode (QCLK = iCLK, QDIR = iDIR) 1h = Direction-count mode (QCLK = xCLK, QDIR = xDIR) 2h = UP count mode for frequency measurement (QCLK = xCLK, QDIR = 1) 3h = DOWN count mode for frequency measurement (QCLK = xCLK, QDIR = 0)
13	SOEN	R/W	0h	Sync output-enable 0h = Disable position-compare sync output 1h = Enable position-compare sync output
12	SPSEL	R/W	0h	Sync output pin selection 0h = Index pin is used for sync output 1h = Strobe pin is used for sync output
11	XCR	R/W	0h	External clock rate 0h = 2x resolution: Count the rising/falling edge 1h = 1x resolution: Count the rising edge only
10	SWAP	R/W	0h	Swap quadrature clock inputs. This swaps the input to the quadrature decoder, reversing the counting direction. 0h = Quadrature-clock inputs are not swapped 1h = Quadrature-clock inputs are swapped
9	IGATE	R/W	0h	Index pulse gating option 0h = Disable gating of Index pulse 1h = Gate the index pin with strobe
8	QAP	R/W	0h	QEPA input polarity 0h = No effect 1h = Negates QEPA input
7	QBP	R/W	0h	QEPB input polarity 0h = No effect 1h = Negates QEPB input
6	QIP	R/W	0h	QEPI input polarity 0h = No effect 1h = Negates QEPI input
5	QSP	R/W	0h	QEPS input polarity 0h = No effect 1h = Negates QEPS input
4-0	RESERVED	R	0h	

15.4.3.13 QEPCTL Register (offset = 2Ah) [reset = 0h]

QEPCTL is shown in [Figure 15-162](#) and described in [Table 15-134](#).

Figure 15-162. QEPCTL Register

15	14	13	12	11	10	9	8
FREE_SOFT		PCRM		SEI		IEI	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
SWI	SEL	IEL		PHEN	QCLM	UTE	WDE
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-134. QEPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation Control Bits. In the values 0 through 3 listed below, x is different for the four following behaviors. QPOSCNT behavior, x refers to the Position counter. QWDTMR behavior, x refers to the Watchdog counter. QUTMR behavior, x refers to the Unit timer. QCTMR behavior, x refers to the Capture timer. 0h = x stops immediately. For QPOSCNT behavior, the stop is on emulation suspend. 1h = x continues to count until the rollover. 2h = x is unaffected by emulation suspend. 3h = x is unaffected by emulation suspend.
13-12	PCRM	R/W	0h	Position counter reset mode 0h = Position counter reset on an index event 1h = Position counter reset on the maximum position 2h = Position counter reset on the first index event 3h = Position counter reset on a unit time event
11-10	SEI	R/W	0h	Strobe event initialization of position counter 0h = Does nothing (action disabled) 1h = Does nothing (action disabled) 2h = Initializes the position counter on rising edge of the QEPS signal 3h = Clockwise Direction: Initializes the position counter on the rising edge of QEPS strobe. Counter Clockwise Direction: Initializes the position counter on the falling edge of QEPS strobe
9-8	IEI	R/W	0h	Index event initialization of position counter 0h = Do nothing (action disabled) 1h = Do nothing (action disabled) 2h = Initializes the position counter on the rising edge of the QEPI signal (QPOSCNT = QPOSINIT) 3h = Initializes the position counter on the falling edge of QEPI signal (QPOSCNT = QPOSINIT)
7	SWI	R/W	0h	Software initialization of position counter 0h = Do nothing (action disabled) 1h = Initialize position counter, this bit is cleared automatically
6	SEL	R/W	0h	Strobe event latch of position counter 0h = The position counter is latched on the rising edge of QEPS strobe (QPOSSLAT = POSCCNT). Latching on the falling edge can be done by inverting the strobe input using the QSP bit in the QDECCTL register. 1h = Clockwise Direction: Position counter is latched on rising edge of QEPS strobe. Counter Clockwise Direction: Position counter is latched on falling edge of QEPS strobe.

Table 15-134. QEPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	IEL	R/W	0h	Index event latch of position counter (software index marker) 0h = Reserved 1h = Latches position counter on rising edge of the index signal 2h = Latches position counter on falling edge of the index signal 3h = Software index marker. Latches the position counter and quadrature direction flag on index event marker. The position counter is latched to the QPOSILAT register and the direction flag is latched in the QEPSTS[QDLF] bit. This mode is useful for software index marking.
3	PHEN	R/W	0h	Quadrature position counter enable/software reset 0h = Reset the eQEP peripheral internal operating flags/read-only registers. Control/configuration registers are not disturbed by a software reset. 1h = eQEP position counter is enabled
2	QCLM	R/W	0h	eQEP capture latch mode 0h = Latch on position counter read by CPU. Capture timer and capture period values are latched into QCTMRLAT and QCPRDLAT registers when CPU reads the QPOSCNT register. 1h = Latch on unit time out. Position counter, capture timer and capture period values are latched into QPOSLAT, QCTMRLAT and QCPRDLAT registers on unit time out.
1	UTE	R/W	0h	eQEP unit timer enable 0h = Disable eQEP unit timer 1h = Enable unit timer
0	WDE	R/W	0h	eQEP watchdog enable 0h = Disable the eQEP watchdog timer 1h = Enable the eQEP watchdog timer

15.4.3.14 QCAPCTL Register (offset = 2Ch) [reset = 0h]

 QCAPCTL is shown in [Figure 15-163](#) and described in [Table 15-135](#).

Figure 15-163. QCAPCTL Register

15	14	13	12	11	10	9	8
CEN	RESERVED						
R/W-0h				R-0h			
7	6	5	4	3	2	1	0
RESERVED	CCPS			UPPS			
R-0h		R/W-0h			R/W-0h		

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-135. QCAPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CEN	R/W	0h	Enable eQEP capture 0h = eQEP capture unit is disabled 1h = eQEP capture unit is enabled
14-7	RESERVED	R	0h	
6-4	CCPS	R/W	0h	eQEP capture timer clock prescaler 0h = CAPCLK = SYSCLKOUT/1 1h = CAPCLK = SYSCLKOUT/2 2h = CAPCLK = SYSCLKOUT/4 3h = CAPCLK = SYSCLKOUT/8 4h = CAPCLK = SYSCLKOUT/16 5h = CAPCLK = SYSCLKOUT/32 6h = CAPCLK = SYSCLKOUT/64 7h = CAPCLK = SYSCLKOUT/128
3-0	UPPS	R/W	0h	Unit position event prescaler 0h = UPEVNT = QCLK/1 1h = UPEVNT = QCLK/2 2h = UPEVNT = QCLK/4 3h = UPEVNT = QCLK/8 4h = UPEVNT = QCLK/16 5h = UPEVNT = QCLK/32 6h = UPEVNT = QCLK/64 7h = UPEVNT = QCLK/128 8h = UPEVNT = QCLK/256 9h = UPEVNT = QCLK/512 Ah = UPEVNT = QCLK/1024 Bh = UPEVNT = QCLK/2048 Ch = Reserved Dh = Reserved Eh = Reserved Fh = Reserved

15.4.3.15 QPOSCTL Register (offset = 2Eh) [reset = 0h]

 QPOSCTL is shown in [Figure 15-164](#) and described in [Table 15-136](#).

Figure 15-164. QPOSCTL Register

15	14	13	12	11	10	9	8
PCSHDW	PCLOAD	PCPOL	PCE	PCSPW			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
PCSPW							
R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-136. QPOSCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	PCSHDW	R/W	0h	Position-compare shadow enable 0h = Shadow disabled, load Immediate 1h = Shadow enabled
14	PCLOAD	R/W	0h	Position-compare shadow load mode 0h = Load on QPOSCNT = 0 1h = Load when QPOSCNT = QPOSCMP
13	PCPOL	R/W	0h	Polarity of sync output 0h = Active HIGH pulse output 1h = Active LOW pulse output
12	PCE	R/W	0h	Position-compare enable/disable 0h = Disable position compare unit 1h = Enable position compare unit
11-0	PCSPW	R/W	0h	Select-position-compare sync output pulse width ... 0h = 1 x 4 x SYSCLKOUT cycles 1h = 2 x 4 x SYSCLKOUT cycles 2h = 3 x 4 x SYSCLKOUT cycles to 4096 x 4 x SYSCLKOUT cycles FFFh = 3 x 4 x SYSCLKOUT cycles to 4096 x 4 x SYSCLKOUT cycles

15.4.3.16 QEINT Register (offset = 30h) [reset = 0h]

 QEINT is shown in [Figure 15-165](#) and described in [Table 15-137](#).

Figure 15-165. QEINT Register

15	14	13	12	11	10	9	8
RESERVED				UTO	IEL	SEL	PCM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-137. QEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	
11	UTO	R/W	0h	Unit time out interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
10	IEL	R/W	0h	Index event latch interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
9	SEL	R/W	0h	Strobe event latch interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
8	PCM	R/W	0h	Position-compare match interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
7	PCR	R/W	0h	Position-compare ready interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
6	PCO	R/W	0h	Position counter overflow interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
5	PCU	R/W	0h	Position counter underflow interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
4	WTO	R/W	0h	Watchdog time out interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
3	QDC	R/W	0h	Quadrature direction change interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
2	PHE	R/W	0h	Quadrature phase error interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
1	PCE	R/W	0h	Position counter error interrupt enable 0h = Interrupt is disabled 1h = Interrupt is enabled
0	RESERVED	R	0h	

15.4.3.17 QFLG Register (offset = 32h) [reset = 0h]

 QFLG is shown in [Figure 15-166](#) and described in [Table 15-138](#).

Figure 15-166. QFLG Register

15	14	13	12	11	10	9	8
RESERVED				UTO	IEL	SEL	PCM
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	INT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-138. QFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	
11	UTO	R	0h	Unit time out interrupt flag 0h = No interrupt generated 1h = Set by eQEP unit timer period match
10	IEL	R	0h	Index event latch interrupt flag 0h = No interrupt generated 1h = This bit is set after latching the QPOSCNT to QPOSILAT
9	SEL	R	0h	Strobe event latch interrupt flag 0h = No interrupt generated 1h = This bit is set after latching the QPOSCNT to QPOSSLAT
8	PCM	R	0h	eQEP compare match event interrupt flag 0h = No interrupt generated 1h = This bit is set on position-compare match
7	PCR	R	0h	Position-compare ready interrupt flag 0h = No interrupt generated 1h = This bit is set after transferring the shadow register value to the active position compare register.
6	PCO	R	0h	Position counter overflow interrupt flag 0h = No interrupt generated 1h = This bit is set on position counter overflow.
5	PCU	R	0h	Position counter underflow interrupt flag 0h = No interrupt generated 1h = This bit is set on position counter underflow.
4	WTO	R	0h	Watchdog timeout interrupt flag 0h = No interrupt generated 1h = Set by watch dog timeout
3	QDC	R	0h	Quadrature direction change interrupt flag 0h = No interrupt generated 1h = This bit is set during change of direction
2	PHE	R	0h	Quadrature phase error interrupt flag 0h = No interrupt generated 1h = Set on simultaneous transition of QEPA and QEPB
1	PCE	R	0h	Position counter error interrupt flag 0h = No interrupt generated 1h = Position counter error
0	INT	R	0h	Global interrupt status flag 0h = No interrupt generated 1h = Interrupt was generated

15.4.3.18 QCLR Register (offset = 34h) [reset = 0h]

QCLR is shown in [Figure 15-167](#) and described in [Table 15-139](#).

Figure 15-167. QCLR Register

15		14		13		12		11		10		9		8	
RESERVED								UTO		IEL		SEL		PCM	
R-0h								R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7		6		5		4		3		2		1		0	
PCR		PCO		PCU		WTO		QDC		PHE		PCE		INT	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-139. QCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	
11	UTO	R/W	0h	Clear unit time out interrupt flag 0h = No effect 1h = Clears the interrupt flag
10	IEL	R/W	0h	Clear index event latch interrupt flag 0h = No effect 1h = Clears the interrupt flag
9	SEL	R/W	0h	Clear strobe event latch interrupt flag 0h = No effect 1h = Clears the interrupt flag
8	PCM	R/W	0h	Clear eQEP compare match event interrupt flag 0h = No effect 1h = Clears the interrupt flag
7	PCR	R/W	0h	Clear position-compare ready interrupt flag 0h = No effect 1h = Clears the interrupt flag
6	PCO	R/W	0h	Clear position counter overflow interrupt flag 0h = No effect 1h = Clears the interrupt flag
5	PCU	R/W	0h	Clear position counter underflow interrupt flag 0h = No effect 1h = Clears the interrupt flag
4	WTO	R/W	0h	Clear watchdog timeout interrupt flag 0h = No effect 1h = Clears the interrupt flag
3	QDC	R/W	0h	Clear quadrature direction change interrupt flag 0h = No effect 1h = Clears the interrupt flag
2	PHE	R/W	0h	Clear quadrature phase error interrupt flag 0h = No effect 1h = Clears the interrupt flag
1	PCE	R/W	0h	Clear position counter error interrupt flag 0h = No effect 1h = Clears the interrupt flag
0	INT	R/W	0h	Global interrupt clear flag 0h = No effect 1h = Clears the interrupt flag and enables further interrupts to be generated if an event flags is set to 1.

15.4.3.19 QFRC Register (offset = 36h) [reset = 0h]

 QFRC is shown in [Figure 15-168](#) and described in [Table 15-140](#).

Figure 15-168. QFRC Register

15	14	13	12	11	10	9	8
RESERVED				UTO	IEL	SEL	PCM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-140. QFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	
11	UTO	R/W	0h	Force unit time out interrupt 0h = No effect 1h = Force the interrupt
10	IEL	R/W	0h	Force index event latch interrupt 0h = No effect 1h = Force the interrupt
9	SEL	R/W	0h	Force strobe event latch interrupt 0h = No effect 1h = Force the interrupt
8	PCM	R/W	0h	Force position-compare match interrupt 0h = No effect 1h = Force the interrupt
7	PCR	R/W	0h	Force position-compare ready interrupt 0h = No effect 1h = Force the interrupt
6	PCO	R/W	0h	Force position counter overflow interrupt 0h = No effect 1h = Force the interrupt
5	PCU	R/W	0h	Force position counter underflow interrupt 0h = No effect 1h = Force the interrupt
4	WTO	R/W	0h	Force watchdog time out interrupt 0h = No effect 1h = Force the interrupt
3	QDC	R/W	0h	Force quadrature direction change interrupt 0h = No effect 1h = Force the interrupt
2	PHE	R/W	0h	Force quadrature phase error interrupt 0h = No effect 1h = Force the interrupt
1	PCE	R/W	0h	Force position counter error interrupt 0h = No effect 1h = Force the interrupt
0	RESERVED	R	0h	

15.4.3.20 QEPSTS Register (offset = 38h) [reset = 0h]

 QEPSTS is shown in [Figure 15-169](#) and described in [Table 15-141](#).

Figure 15-169. QEPSTS Register

15		14		13		12		11		10		9		8	
RESERVED															
R-0h															
7		6		5		4		3		2		1		0	
UPEVNT	FDF	QDF	QDLF	COEF	CDEF	FIMF	PCEF								
R-0h	R-0h	R-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h								

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-141. QEPSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	
7	UPEVNT	R	0h	Unit position event flag 0h = No unit position event detected 1h = Unit position event detected. Write 1 to clear.
6	FDF	R	0h	Direction on the first index marker. Status of the direction is latched on the first index event marker. 0h = Counter-clockwise rotation (or reverse movement) on the first index event 1h = Clockwise rotation (or forward movement) on the first index event
5	QDF	R	0h	Quadrature direction flag 0h = Counter-clockwise rotation (or reverse movement) 1h = Clockwise rotation (or forward movement)
4	QDLF	R	0h	eQEP direction latch flag. Status of direction is latched on every index event marker. 0h = Counter-clockwise rotation (or reverse movement) on index event marker 1h = Clockwise rotation (or forward movement) on index event marker
3	COEF	R/W	0h	Capture overflow error flag 0h = Sticky bit, cleared by writing 1 1h = Overflow occurred in eQEP Capture timer (QEPCTMR)
2	CDEF	R/W	0h	Capture direction error flag 0h = Sticky bit, cleared by writing 1 1h = Direction change occurred between the capture position event.
1	FIMF	R/W	0h	First index marker flag 0h = Sticky bit, cleared by writing 1 1h = Set by first occurrence of index pulse
0	PCEF	R	0h	Position counter error flag. This bit is not sticky and it is updated for every index event. 0h = No error occurred during the last index transition. 1h = Position counter error

15.4.3.21 QCTMR Register (offset = 3Ah) [reset = 0h]

QCTMR is shown in [Figure 15-170](#) and described in [Table 15-142](#).

Figure 15-170. QCTMR Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QCTMR															
R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-142. QCTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCTMR	R/W	0h	This register provides time base for edge capture unit.

15.4.3.22 QCPRD Register (offset = 3Ch) [reset = 0h]

QCPRD is shown in [Figure 15-171](#) and described in [Table 15-143](#).

Figure 15-171. QCPRD Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QCPRD															
R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-143. QCPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCPRD	R/W	0h	This register holds the period count value between the last successive eQEP position events

15.4.3.23 QCTMRLAT Register (offset = 3Eh) [reset = 0h]

QCTMRLAT is shown in [Figure 15-172](#) and described in [Table 15-144](#).

Figure 15-172. QCTMRLAT Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QCTMRLAT															
R-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-144. QCTMRLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCTMRLAT	R	0h	The eQEP capture timer value can be latched into this register on two events, that is, unit timeout event, reading the eQEP position counter.

15.4.3.24 QCPRDLAT Register (offset = 40h) [reset = 0h]

 QCPRDLAT is shown in [Figure 15-173](#) and described in [Table 15-145](#).

Figure 15-173. QCPRDLAT Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QCPRDLAT															
R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-145. QCPRDLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCPRDLAT	R/W	0h	eQEP capture period value can be latched into this register on two events, that is, unit timeout event, reading the eQEP position counter.

15.4.3.25 REVID Register (offset = 5Ch) [reset = 44D31103h]

 REVID is shown in [Figure 15-174](#) and described in [Table 15-146](#).

Figure 15-174. REVID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REV																															
R-44D31103h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -n = value after reset

Table 15-146. REVID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REV	R	44D31103h	eQEP revision ID

Universal Serial Bus (USB)

This chapter describes the USB of the device.

Topic	Page
16.1 Integration	2559
16.2 Functional Description	2562
16.3 Supported Use Cases	2625
16.4 USB Registers	2626