

Micro Processor & Controller

Pulse Wide Modulation
&
H-Bridge

Pulse Width Modulation - PWM

A Pulse Width Modulation (PWM) Signal is a method for generating an analog signal using a digital source. A PWM signal consists of two main components that define its behavior: a duty cycle and a frequency. The duty cycle describes the amount of time the signal is in a **ON** state as a percentage of the **TOTAL** time of it takes to complete one cycle. The frequency determines how fast the PWM completes a cycle, and therefore how fast it switches between on and off states. By cycling a digital signal off and on at a fast enough rate, and with a certain duty cycle, the output will appear to behave like an average constant analog signal when providing power to devices.

PWM Applications

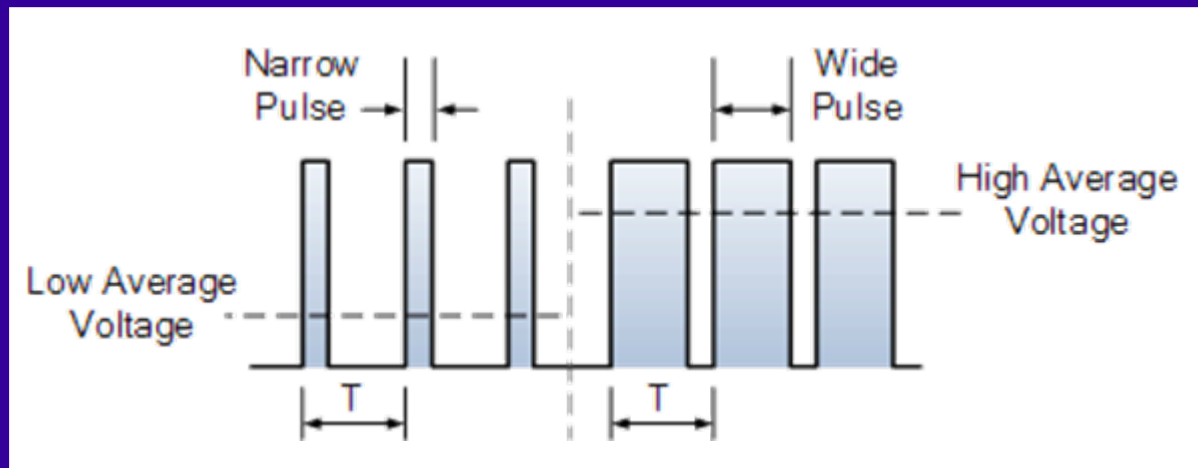
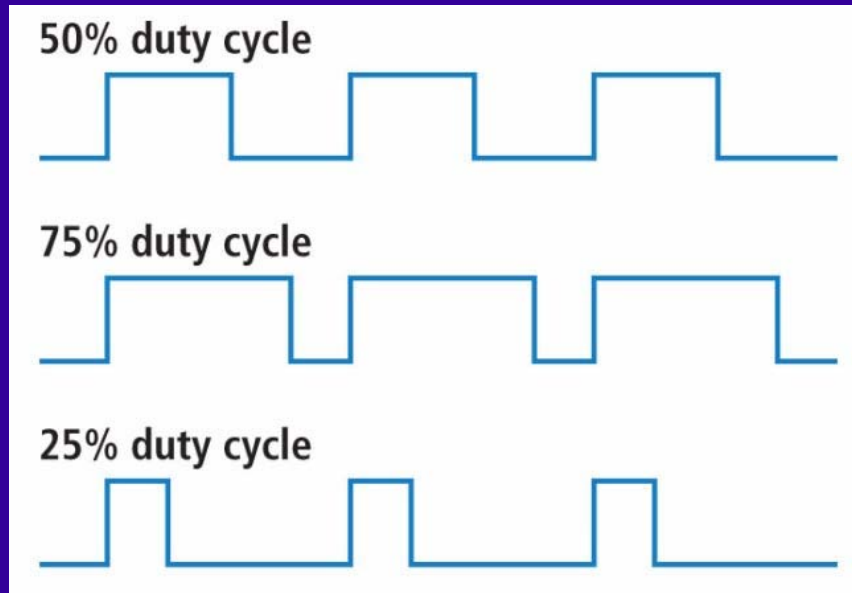
PWM signals are used for a wide variety of control applications. Their main use is for controlling DC motors but it can also be used to control valves, pumps, hydraulics, and other mechanical and power parts. The frequency that the PWM signal needs to be set at will be dependent on the application and the response time of the system that is being powered. Below are a few applications and some typical minimum PWM frequencies required:

- Heating elements: 10-100 Hz or higher
- DC electric motors: 5-10 kHz or higher
- Power supplies & converter: 20-200 kHz or higher

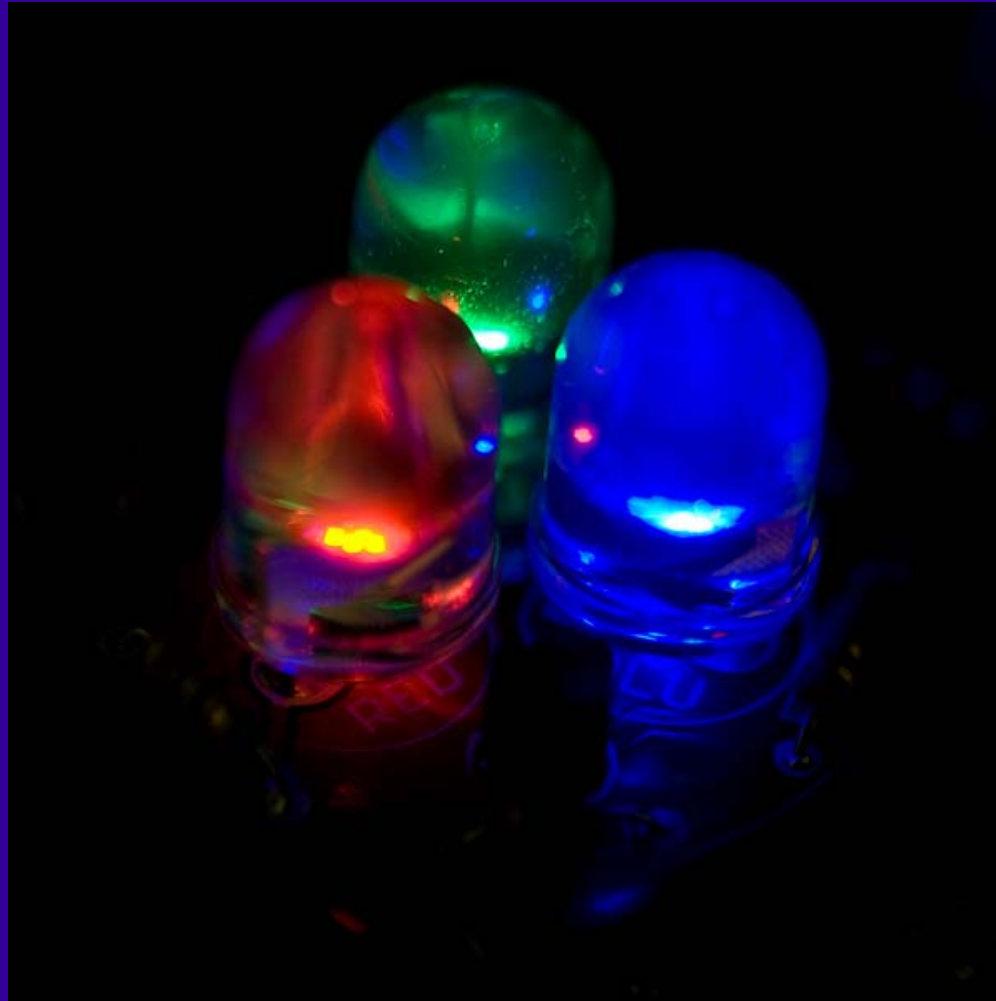
Robotic Claw Controlled PWM



Duty Cycle

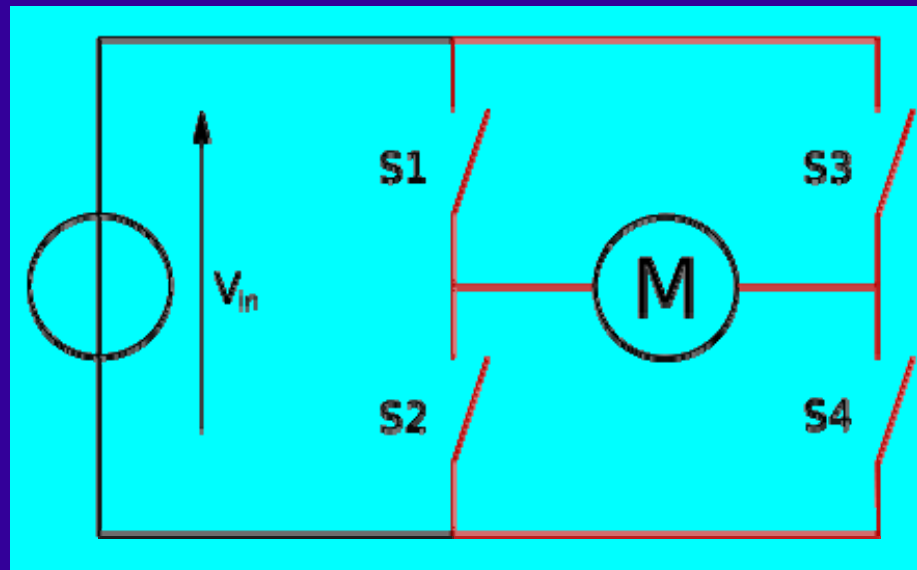


PWM Used to Control LED Brightness

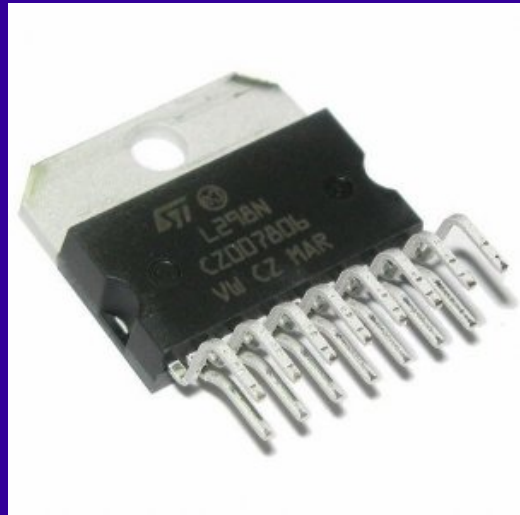


H-Bridge

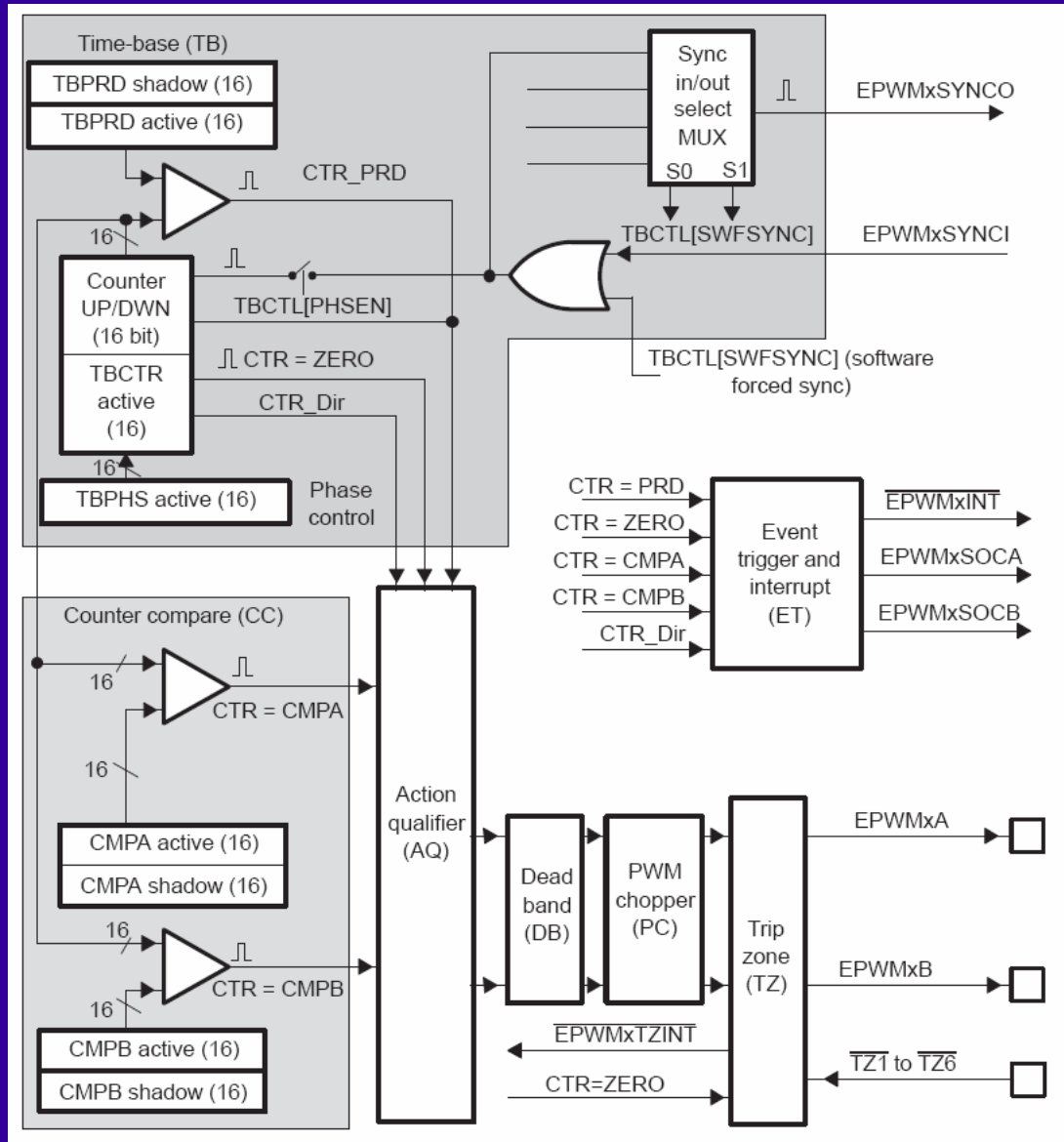
A H bridge is an electronic circuit that enables a voltage or current to be applied across a load in either direction. These circuits are often used in robotics and other applications to allow DC motors to run forwards and backwards



H-Bridge L298 & L293

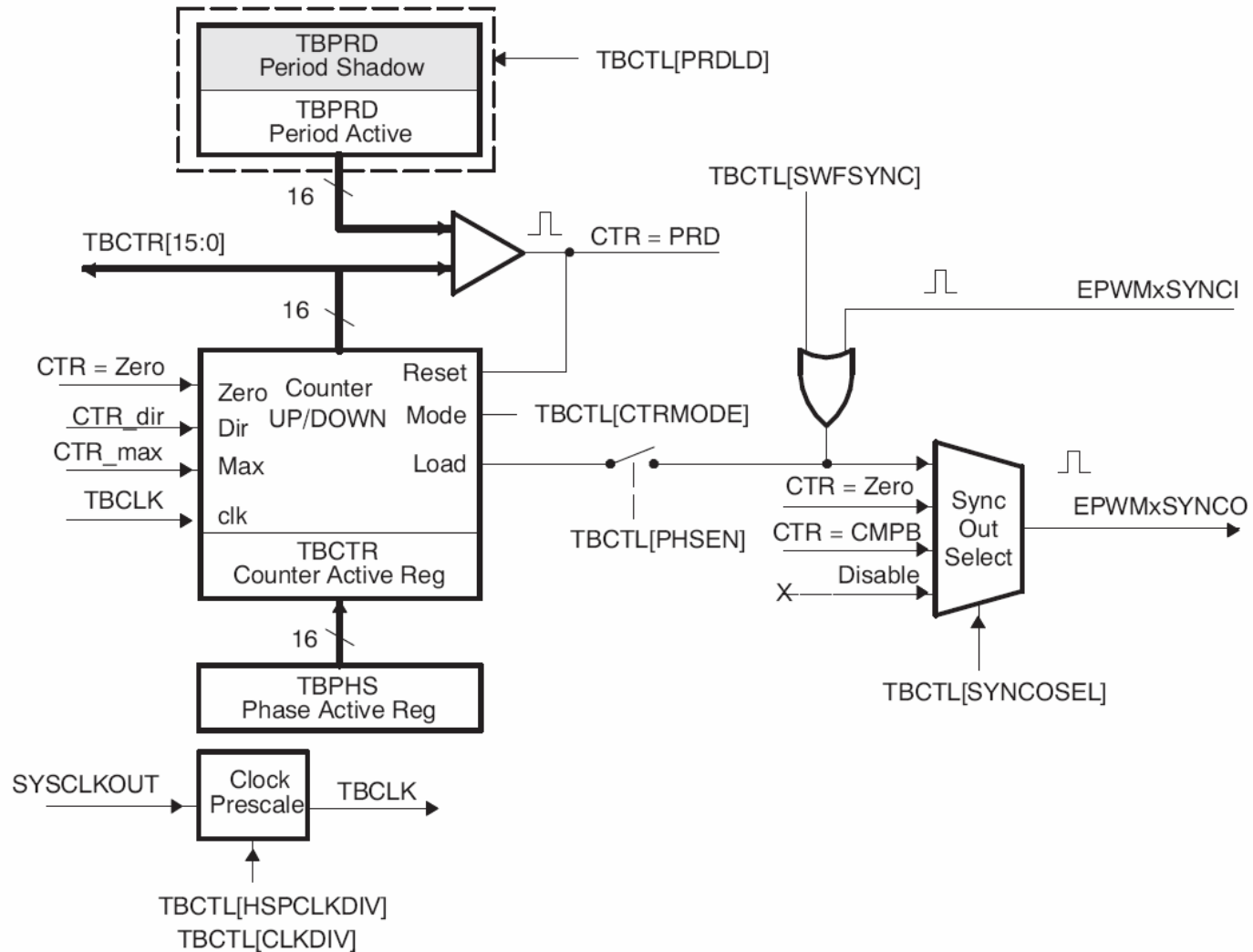


Delfino PWM Modules (Dual A&B 1-4)



Time Base Sub-Module

Figure 2-2. Time-Base Submodule Signals and Registers



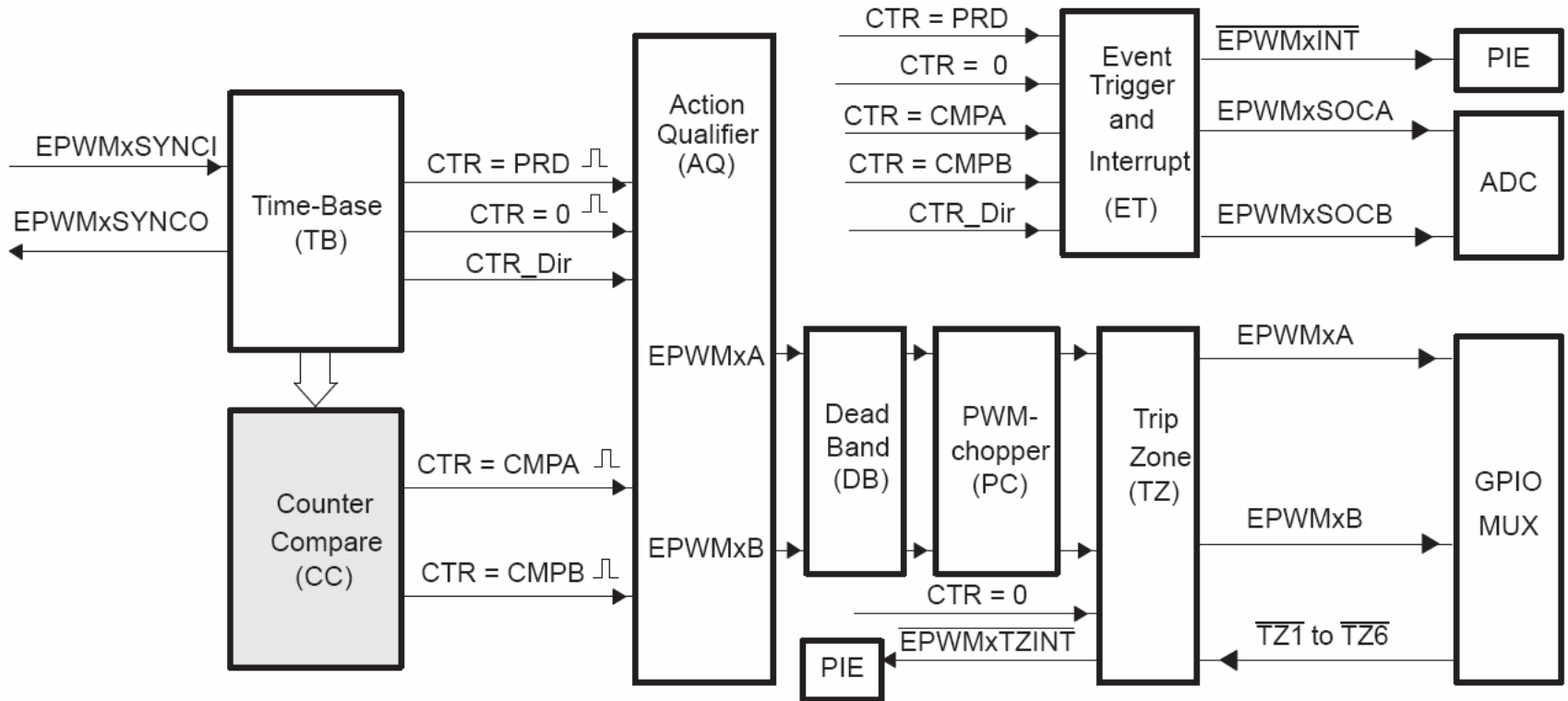
Time Base Sub-Module Registers

Table 2-2. Time-Base Submodule Registers

Register	Address offset	Shadowed	Description
TBCTL	0x0000	No	Time-Base Control Register
TBSTS	0x0001	No	Time-Base Status Register
TBPHSHR	0x0002	No	HRPWM extension Phase Register ⁽¹⁾
TBPHS	0x0003	No	Time-Base Phase Register
TBCTR	0x0004	No	Time-Base Counter Register
TBPRD	0x0005	Yes	Time-Base Period Register

Counter Compare Sub-Module

Figure 2-11. Counter-Compare Submodule



Counter Compare Sub-Module Registers

Table 2-4. Counter-Compare Submodule Registers

Register Name	Address Offset	Shadowed	Description
CMPCTL	0x0007	No	Counter-Compare Control Register.
CMPAHR	0x0008	Yes	HRPWM Counter-Compare A Extension Register ⁽¹⁾
CMPA	0x0009	Yes	Counter-Compare A Register
CMPB	0x000A	Yes	Counter-Compare B Register

Action-Qualifier Sub-Module

2.4.1 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 (TBCTR = TBPRD).
 - CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000)
 - CTR = CMPA: Time-base counter equal to the counter-compare A register (TBCTR = CMPA)
 - CTR = CMPB: Time-base counter equal to the counter-compare B register (TBCTR = 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. .

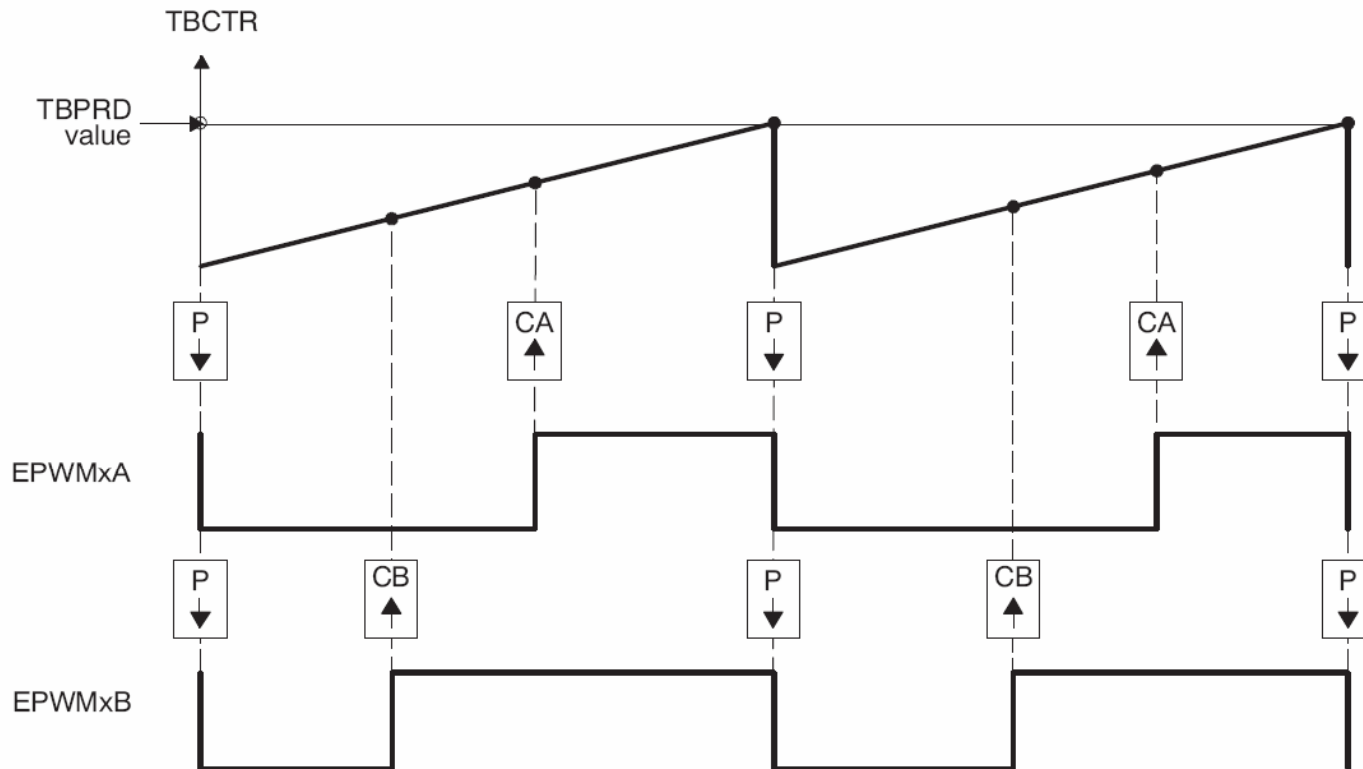
Action-Qualifier Events

Figure 2-19. Possible Action-Qualifier Actions for EPWMxA and EPWMxB Outputs

S/W force	TB Counter equals:				Actions
	Zero	Comp A	Comp B	Period	
SW X	Z X	CA X	CB X	P X	Do Nothing
SW ↓	Z ↓	CA ↓	CB ↓	P ↓	Clear Low
SW ↑	Z ↑	CA ↑	CB ↑	P ↑	Set High
SW T	Z T	CA T	CB T	P T	Toggle

Action-Qualifier Example

Figure 2-22. Up, Single Edge Asymmetric Waveform With Independent Modulation on EPWMxA and EPWMxB—Active Low



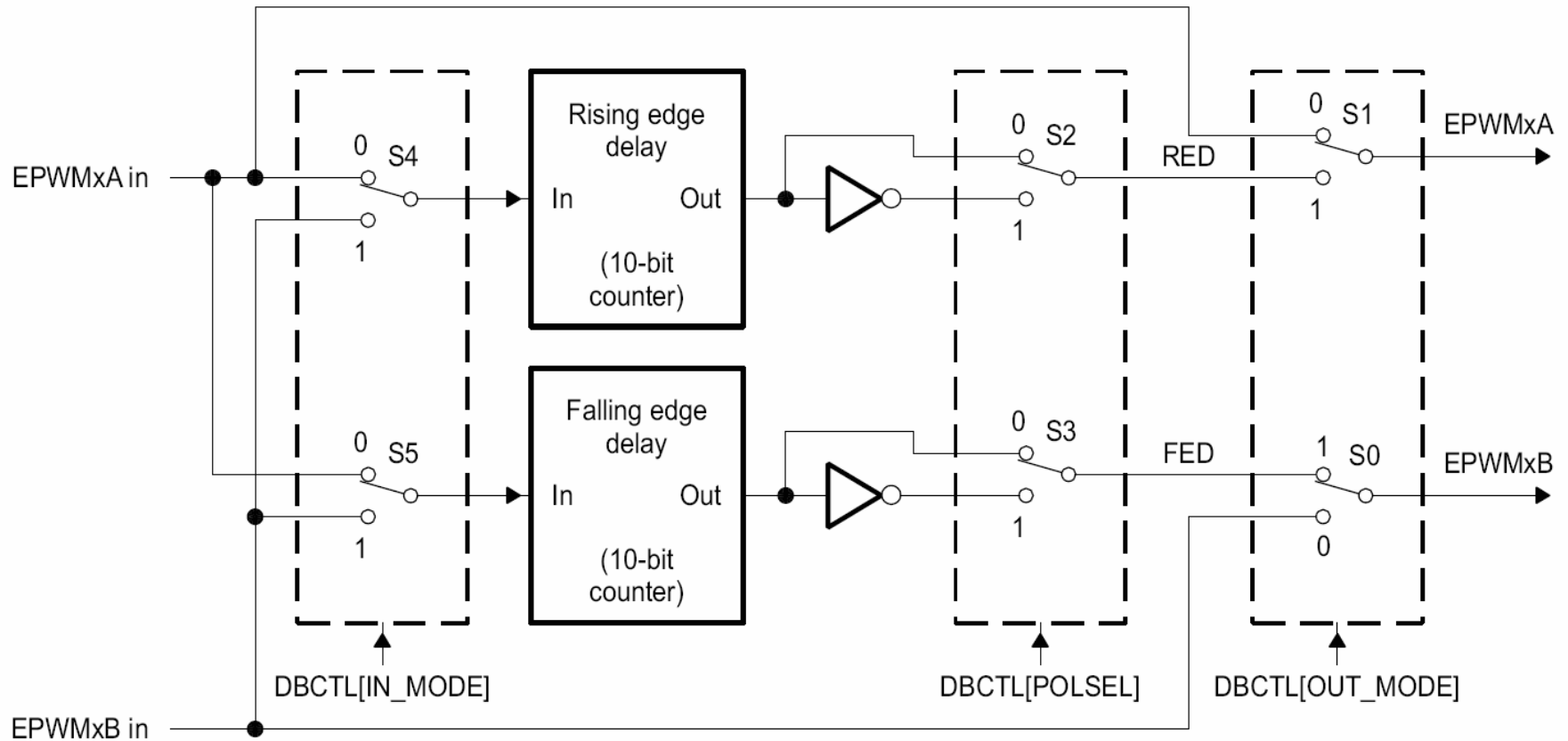
- A PWM period = $(TBPRD + 1) \times T_{TBCLK}$
- B Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).
- C Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).
- D The Do Nothing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

Action-Qualifier Example Code

```
// Initialization Time
// = = = = =
EPwm1Regs.TBPRD = 600; // Period = 601 TBCLK counts
EPwm1Regs.CMPA.half.CMPA = 350; // Compare A = 350 TBCLK counts
EPwm1Regs.CMPB = 200; // Compare B = 200 TBCLK counts
EPwm1Regs.TBPHS = 0; // Set Phase register to zero
EPwm1Regs.TBCTR = 0; // clear TB counter
EPwm1Regs.TBCTL.bit.CTRMODE = TB_UP;
EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE; // Phase loading disabled
EPwm1Regs.TBCTL.bit.PRDLN = TB_SHADOW;
EPwm1Regs.TBCTL.bit.SYNCSEL = TB_SYNC_DISABLE;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1; // TBCLK = SYSCLKOUT
EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO; // load on TBCTR = Zero
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO; // load on TBCTR = Zero
EPwm1Regs.AQCTLA.bit.PRD = AQ_CLEAR;
EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm1Regs.AQCTLB.bit.PRD = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.CBU = AQ_SET;
//
// Run Time
// = = = = =
EPwm1Regs.CMPA.half.CMPA = Duty1A; // adjust duty for output EPWM1A
EPwm1Regs.CMPB = Duty1B; // adjust duty for output EPWM1B
```

Dead Band Sub-Module

Figure 2-28. Configuration Options for the Dead-Band Submodule



Dead Band Sub-Module

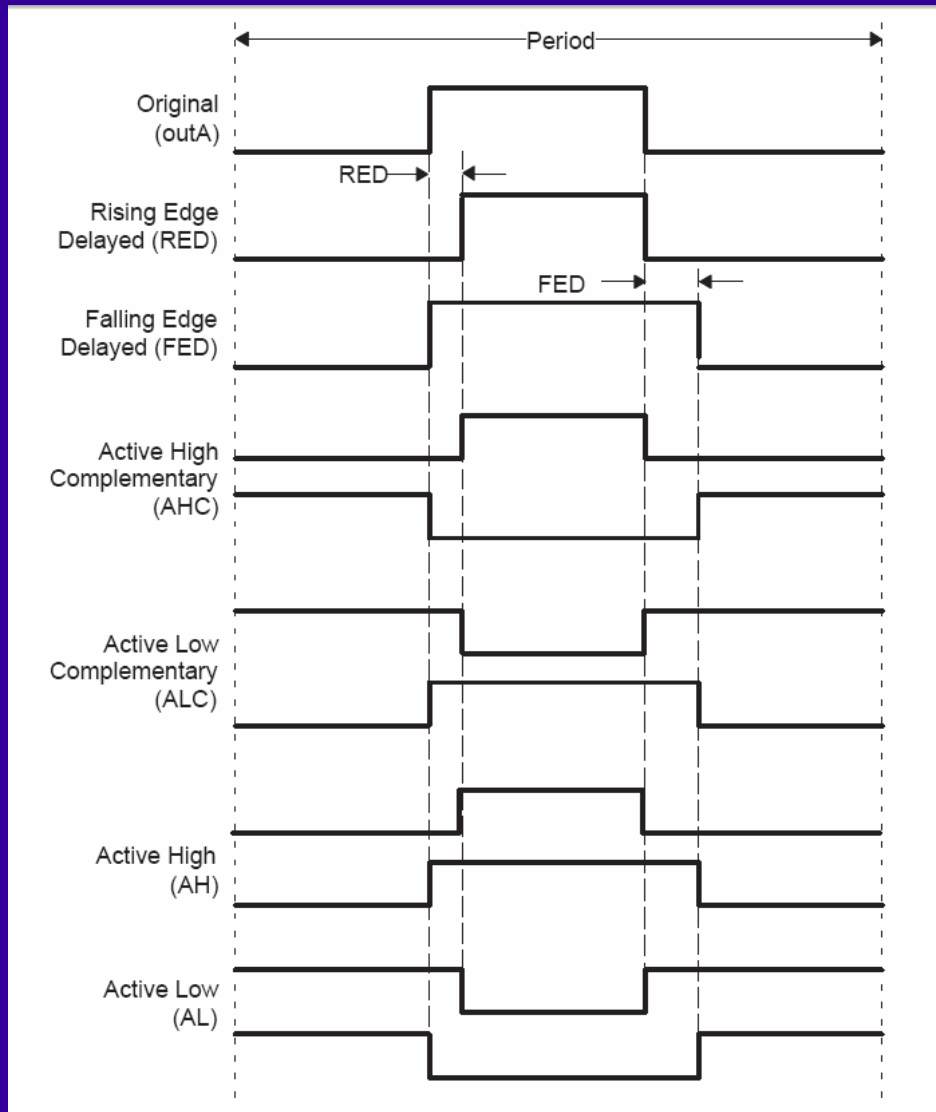
2.5.1 Purpose of the Dead-Band Submodule

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 submodule described here should be used.

The key functions of the dead-band module 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)

Dead Band Modes



Flax Init PWM

```
// This function initializes ePWM4
void FlaxInitEPwm4(Uint16 Period)
{
#define EPWM4_CMPA_INIT 0
#define EPWM4_CMPB_INIT 0

    EALLOW;
    // Time-Base Submodule Register
    EPwm4Regs.TBPRD = Period;           // Set Timer Period
    EPwm4Regs.TBPHS.all = 0;           // Set Timer Phase
    EPwm4Regs.TBCTR = 0x0000;          // Clear counter

    EPwm4Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;           // Count up
    EPwm4Regs.TBCTL.bit.PHSEN = TB_DISABLE;              // Disable phase loading
    EPwm4Regs.TBCTL.bit.PRDLT = TB_SHADOW;               // Load Period from Shadows
    EPwm4Regs.TBCTL.bit.SYNCSEL = TB_SYNC_IN;           // Synchronization Out Select
    EPwm4Regs.TBCTL.bit.HSPCLKDIV = TB_HSClk_DIV1;      // PWM Clock = CLK /
    (HSPCLKDIV*CLKDIV)
    EPwm4Regs.TBCTL.bit.CLKDIV = TB_CLK_DIV1;           // PWM Freq = PWMClock / Period
    EPwm4Regs.TBCTL.bit.FREE_SOFT = FREESOFT;          // Set Emulation mode to free run

    // Counter Compare Submodule Register
    EPwm4Regs.CMPA.half.CMPA = 0;           // Set compare A value
    EPwm4Regs.CMPB = 0;                     // Set compare B value
}
```

See the full example in the course site