TMS320x28xx, 28xxx Enhanced Capture (eCAP) Module

Reference Guide

Literature Number: SPRU807A

November 2004 – Revised October 2006

TEXAS INSTRUMENTS

Contents

List of Figures

List of Tables

Read This First

The enhanced capture (eCAP) module is used in systems where accurate timing of external events is important. This guide describes the module and how to use it.

Related Documentation From Texas Instruments

The following documents describe the TMS320C28x and related support tools and can be downloaded from the TI website (www.ti.com):

Data Manuals —

SPRS230: —TMS320F2809, F2808, F2806, F2802, F2801, F2801x UCD9501, C2802, C2801 DSPs Data Manual contains the pinout, signal descriptions, as well as electrical and timing specifications for the F280x devices.

SPRS357: —TMS320F28044 Digital Signal Processor Data Manual contains the pinout, signal descriptions, as well as electrical and timing specifications for the F28044 device.

User's Guides —

SPRU051: —TMS320x28xx, 28xxx Serial Communication Interface (SCI) Reference Guide describes the SCI, which is a two-wire asynchronous serial port, commonly known as a UART. The SCI modules support digital communications between the CPU and other asynchronous peripherals that use the standard non-return-to-zero (NRZ) format.

SPRU059: —TMS320x28xx, 28xxx Serial Peripheral Interface (SPI) Reference Guide describes the SPI - a high-speed synchronous serial input/output (I/O) port - that allows a serial bit stream of programmed length (one to sixteen bits) to be shifted into and out of the device at a programmed bit-transfer rate.

SPRU074: —TMS320x28xx, 28xxx Enhanced Controller Area Network (eCAN) Reference Guide describes the eCAN that uses established protocol to communicate serially with other controllers in electrically noisy environments.

SPRU430: —TMS320C28x DSP CPU and Instruction Set Reference Guide describes the central processing unit (CPU) and the assembly language instructions of the TMS320C28x fixed-point digital signal processors (DSPs). It also describes emulation features available on these DSPs.

SPRU513: — TMS320C28x Assembly Language Tools User's Guide describes the assembly language tools (assembler and other tools used to develop assembly language code), assembler directives, macros, common object file format, and symbolic debugging directives for the TMS320C28x device.

SPRU514: —TMS320C28x Optimizing C Compiler User's Guide describes the TMS320C28x™ C/C++ compiler. This compiler accepts ANSI standard C/C++ source code and produces TMS320 DSP assembly language source code for the TMS320C28x device.

SPRU566: —TMS320x28xx, 28xxx Peripheral Reference Guide describes the peripheral reference guides of the 28x digital signal processors (DSPs).

SPRU608: — The TMS320C28x Instruction Set Simulator Technical Overview describes the simulator, available within the Code Composer Studio for TMS320C2000 IDE, that simulates the instruction set of the C28x™ core.

SPRU625: —TMS320C28x DSP/BIOS Application Programming Interface (API) Reference Guide describes development using DSP/BIOS.

SPRU712: —TMS320x28xx, 28xxx System Control and Interrupts Reference Guide describes the various interrupts and system control features of the 280x digital signal processors (DSPs).

SPRU716: —TMS320x280x, 2801x, 2804x Analog-to-Digital Converter (ADC) Reference Guide describes how to configure and use the on-chip ADC module, which is a 12-bit pipelined ADC.

SPRU721: — TMS320x28xx, 28xxx Inter-Integrated Circuit (I2C) Reference Guide describes the features and operation of the inter-integrated circuit (I^2C) module that is available on the TMS320x280x digital signal processor (DSP).

SPRU722: —TMS320x280x, 2801x, 2804x Boot ROM Reference Guide describes the purpose and features of the bootloader (factory-programmed boot-loading software). It also describes other contents of the device on-chip boot ROM and identifies where all of the information is located within that memory.

SPRU790: —TMS320x28xx, 28xxx Enhanced Quadrature Encoder Pulse (eQEP) Reference Guide describes the eQEP module, which is used for interfacing with a linear or rotary incremental encoder to get position, direction, and speed information from a rotating machine in high performance motion and position control systems. It includes the module description and registers

SPRU791: —TMS320x28xx, 28xxx Enhanced Pulse Width Modulator (ePWM) Module Reference Guide describes the main areas of the enhanced pulse width modulator that include digital motor control, switch mode power supply control, UPS (uninterruptible power supplies), and other forms of power conversion

SPRU807: —TMS320x28xx, 28xxx Enhanced Capture (eCAP) Module Reference Guide describes the enhanced capture module. It includes the module description and registers.

SPRU924: —TMS320x28xx, 28xxx High-Resolution Pulse Width Modulator (HRPWM) describes the operation of the high-resolution extension to the pulse width modulator (HRPWM)

Application Reports —

SPRAA58: — TMS320x281x to TMS320x280x Migration Overview describes differences between the Texas Instruments TMS320x281x and TMS320x280x DSPs to assist in application migration from the 281x to the 280x. While the main focus of this document is migration from 281x to 280x, users considering migrating in the reverse direction (280x to 281x) will also find this document useful.

SPRA550: —3.3 V DSP for Digital Motor Control describes a scenario of a 3.3-V-only motor controller indicating that for most applications, no significant issue of interfacing between 3.3 V and 5 V exists. On-chip 3.3-V analog-to-digital converter (ADC) versus 5-V ADC is also discussed. Guidelines for component layout and printed circuit board (PCB) design that can reduce system noise and EMI effects are summarized.

SPRA820: —Online Stack Overflow Detection on the TMS320C28x DSP presents the methodology for online stack overflow detection on the TMS320C28x™ DSP. C-source code is provided that contains functions for implementing the overflow detection on both DSP/BIOS™ and non-DSP/BIOS applications.

SPRA861: —RAMDISK: A Sample User-Defined C I/O Driver provides an easy way to use the sophisticated buffering of the high-level CIO functions on an arbitrary device. This application report presents a sample implementation of a user-defined device driver.

SPRA873: —Thermo-Electric Cooler Control Using a TMS320F2812 DSP & DRV592 Power Amplifier presents a thermoelectric cooler system consisting of a Texas Instruments TMS320F2812 digital signal processor (DSP) and DRV592 power amplifier. The DSP implements a digital proportional-integral-derivative feedback controller using an integrated 12-bit analog-to-digital converter to read the thermistor, and direct output of pulse-width-modulated waveforms to the H-bridge DRV592 power amplifier. A complete description of the experimental system, along with software and software operating instructions, is provided.

SPRA876: —Programming Examples for the TMS320F281x eCAN contains several programming examples to illustrate how the eCAN module is set up for different modes of operation to help you come up to speed quickly in programming the eCAN. All projects and CANalyzer configuration files are included in the attached SPRA876.zip file.

SPRA953: —IC Package Thermal Metrics describes the traditional and new thermal metrics and will put their application in perspective with respect to system level junction temperature estimation.

SPRA958: —Running an Application from Internal Flash Memory on the TMS320F281x DSP (Rev. B) covers the requirements needed to properly configure application software for execution from on-chip flash memory. Requirements for both DSP/BIOS™ and non-DSP/BIOS projects are presented. Example code projects are included.

SPRA963: —Reliability Data for TMS320LF24x and TMS320F281x Devices describes reliability data for TMS320LF24x and TMS320F281x devices.

SPRA989: —F2810, F2811, and F2812 ADC Calibration describes a method for improving the absolute accuracy of the 12-bit analog-to-digital converter (ADC) found on the F2810/F2811/F2812 devices. This application note is accompanied by an example program (ADCcalibration.zip) that executes from RAM on the F2812 eZdsp.

SPRA991: —Simulation Fulfills its Promise for Enhancing Debug and Analysis - A White Paper describes simulation enhancements that enable developers to speed up the development cycle by allowing them to evaluate system alternatives more effectively.

Trademarks

TMS320C28x, C28x are trademarks of Texas Instruments.

Reference Guide
SPRU807A – November 2004 – Revised October 2006

Enhanced Capture (eCAP) Module

The enhanced Capture (eCAP) module is essential in systems where accurate timing of external events is important.

This reference guide is applicable for the eCAP found on the TMS320x28xx and TMS320x28xxx family of processors. This includes all Flash-based, ROM-based, and RAM-based devices within the 280xx and 28xxx family.

1 Introduction

Uses for eCAP include:

- Speed measurements of rotating machinery (e.g., 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

The eCAP module described in this guide includes the following features:

- 32-bit time base with 10-nS time resolution with a 100-MHz system clock
- 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

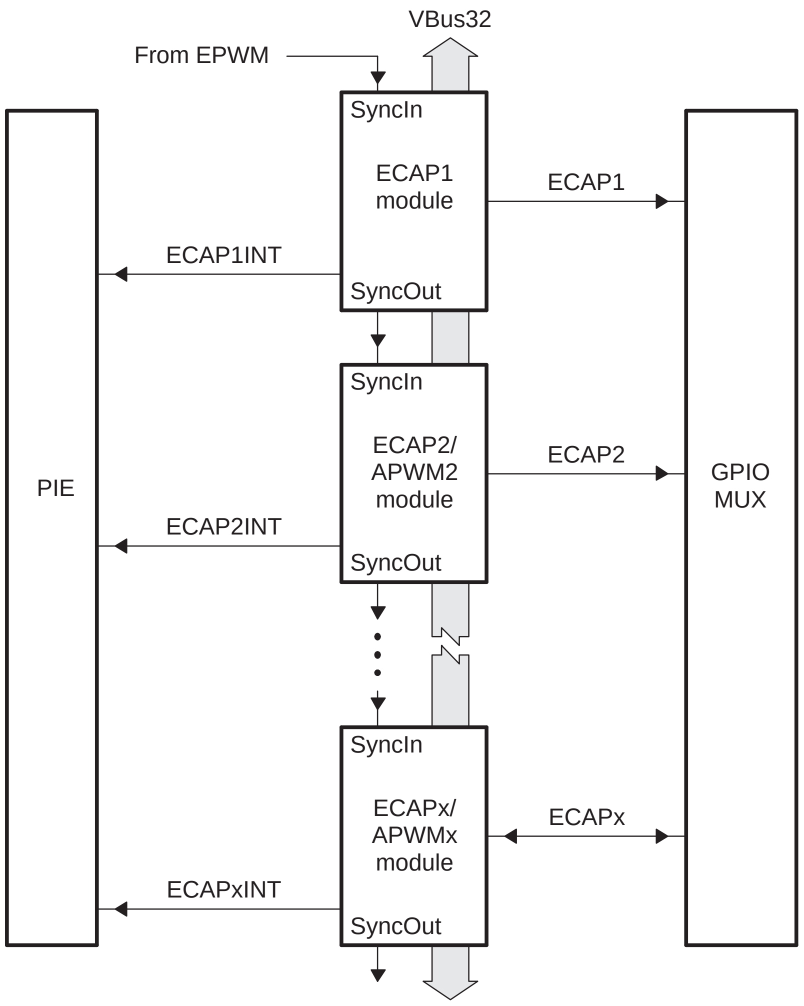
2 Description

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 x 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 280x system as shown in Figure 1. The number of modules is device-dependent and is based on target application needs.

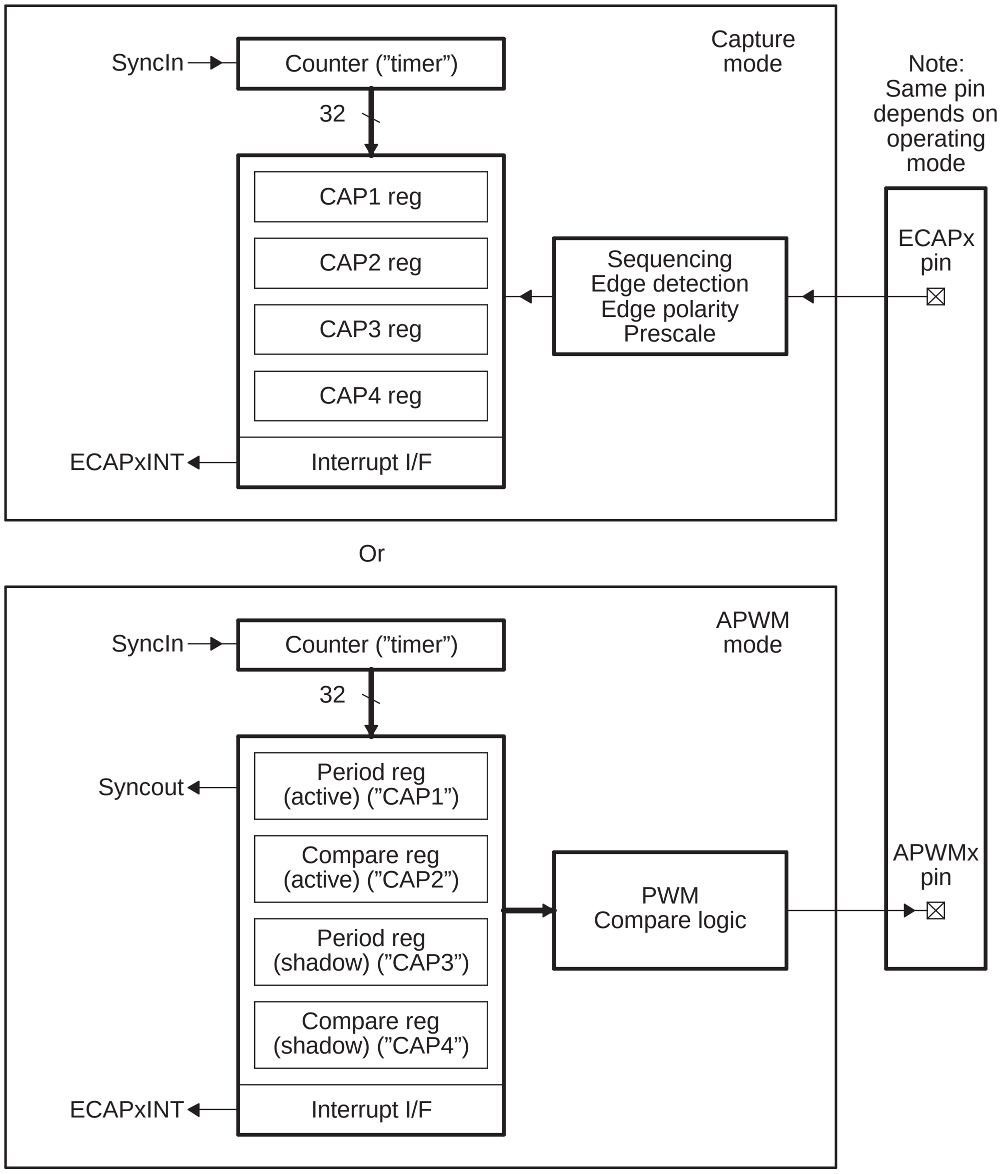
Figure 1. Multiple eCAP Modules In A 28x System



3 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 2 is a high-level view of both the capture and auxiliary pulse-width modulator (APWM) modes of operation.

Figure 2. Capture and APWM Modes of Operation



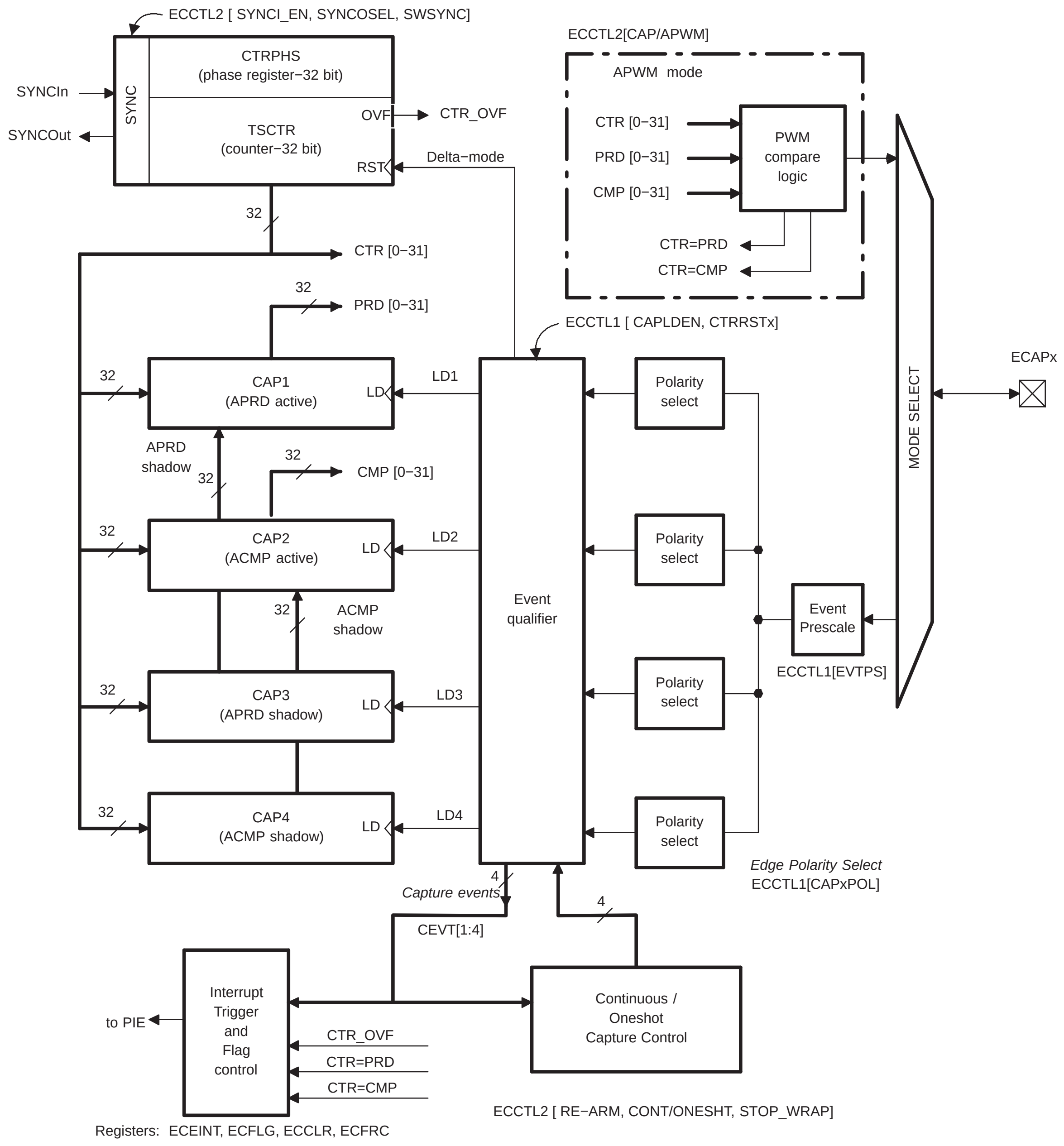
A A single pin is shared between CAP and APWM functions. In capture mode, it is an input; in APWM mode, it is an output.

B 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.

4 Capture Mode Description

Figure 3 shows the various components that implement the capture function.

Figure 3. Capture Function Diagram

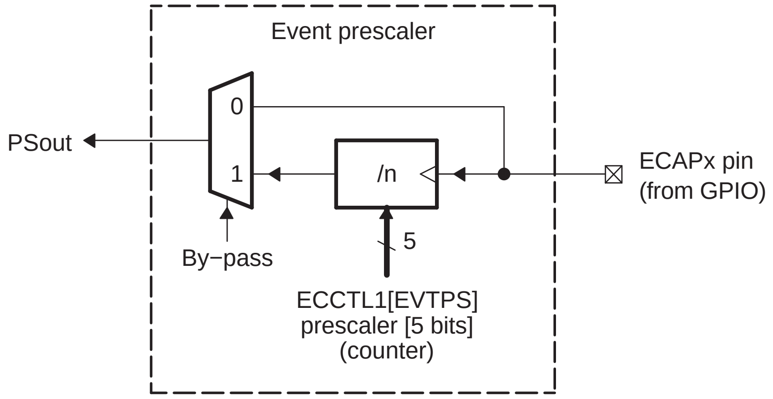


4.1 *Event Prescaler*

- An input capture signal (pulse train) can be prescaled by N = 2-62 (in multiples of 2) or can bypass the prescaler.

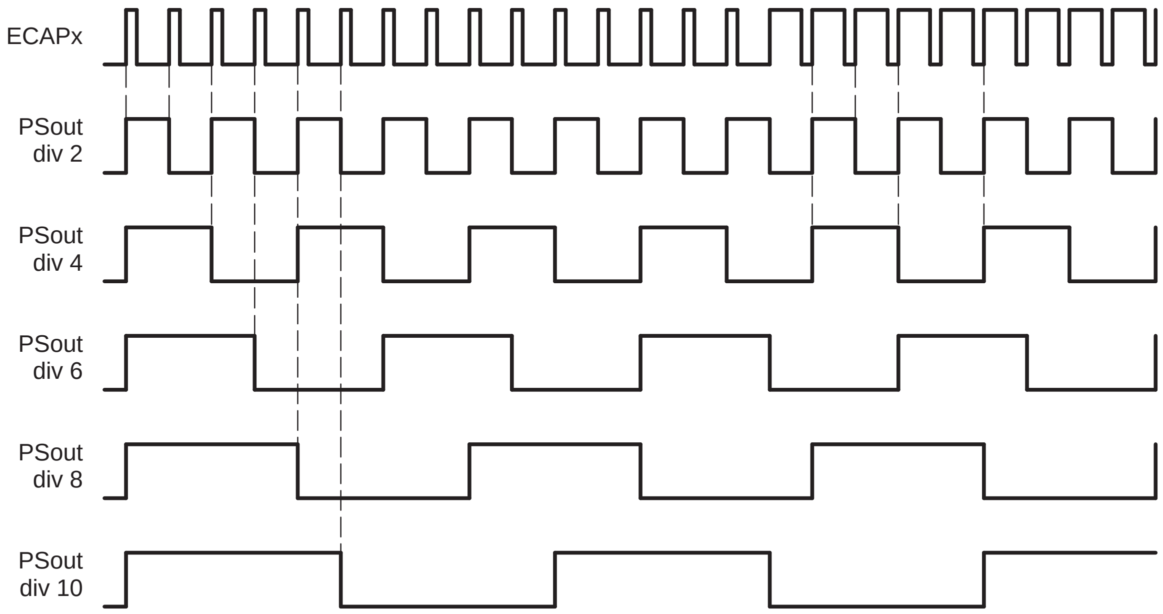
  This is useful when very high frequency signals are used as inputs. Figure 4 shows a functional diagram and Figure 5 shows the operation of the prescale function.

Figure 4. Event Prescale Control



A When a prescale value of 1 is chosen (i.e. ECCTL1[13:9] = 0,0,0,0,0) the input capture signal by-passes the prescale logic completely.

Figure 5. Prescale Function Waveforms



4.2 *Edge Polarity Select and Qualifier*

- Four independent edge polarity (rising edge/falling edge) selection MUXes 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 CAPx register by the Mod4 counter. The CAPx register is loaded on the falling edge.

4.3 *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.

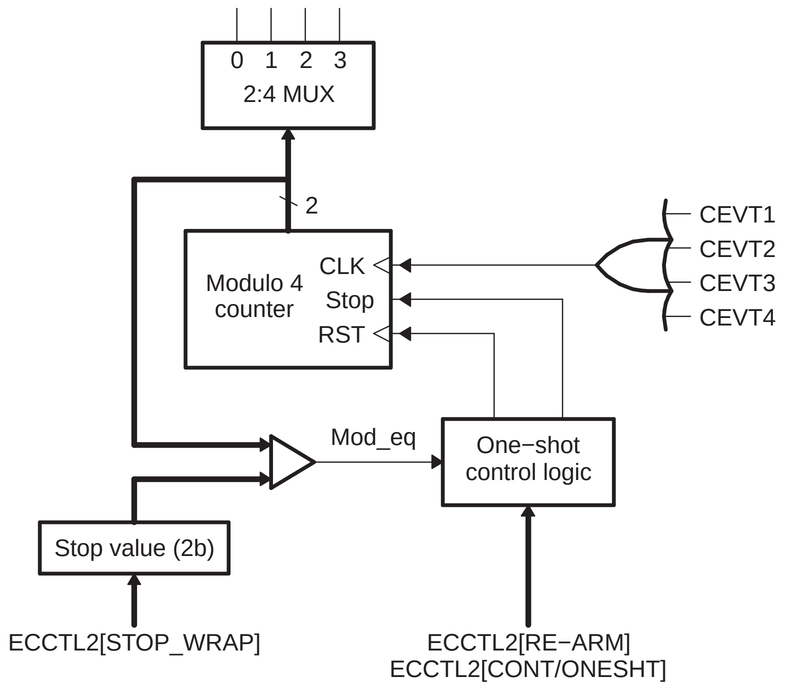
The continuous/one-shot block controls the start/stop and reset (zero) functions of the Mod4 counter via a mono-shot type of action that can be triggered by the stop-value comparator and re-armed via software control.

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 (i.e., 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.

Figure 6. Details of the Continuous/One-shot Block



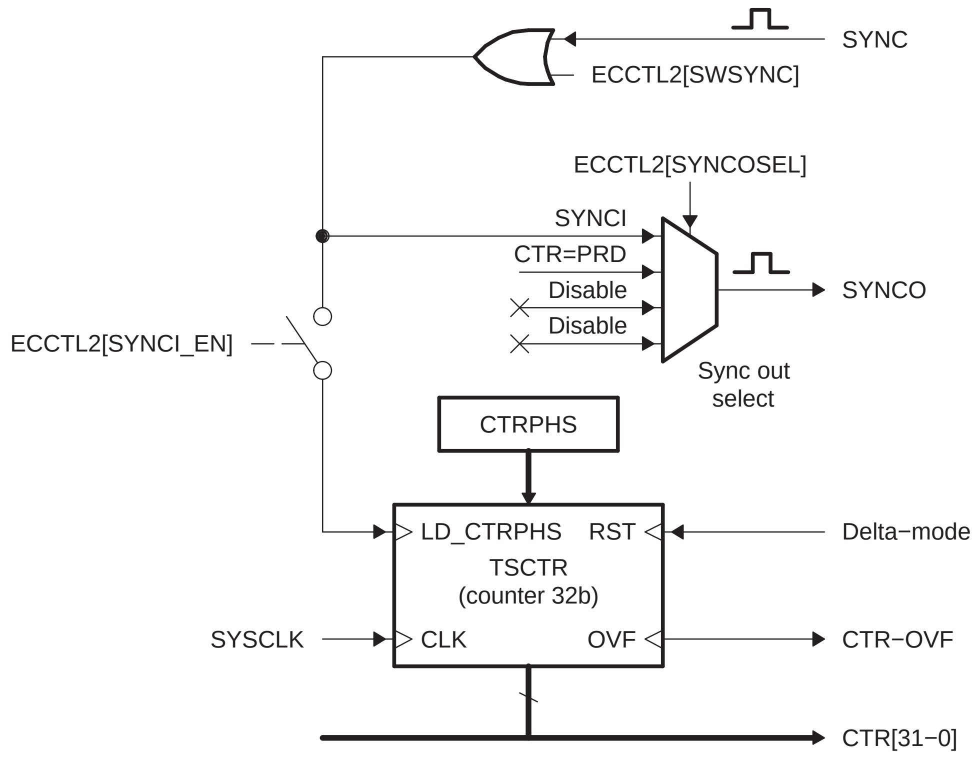
4.4 32-Bit Counter and Phase Control

This counter 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 7. Details of the Counter and Synchronization Block



4.5 CAP1-CAP4 Registers

These 32-bit registers are fed by the 32-bit counter timer bus, CTR[0-31] and are loaded (i.e., 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, i.e. 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.

4.6 Interrupt Control

An Interrupt can be generated on capture events (CEVT1-CEVT4, CTROVF) or APWM events (CTR = PRD, CTR = CMP).

A counter overflow event (FFFFFFFF->00000000) is also provided as an interrupt source (CTROVF).

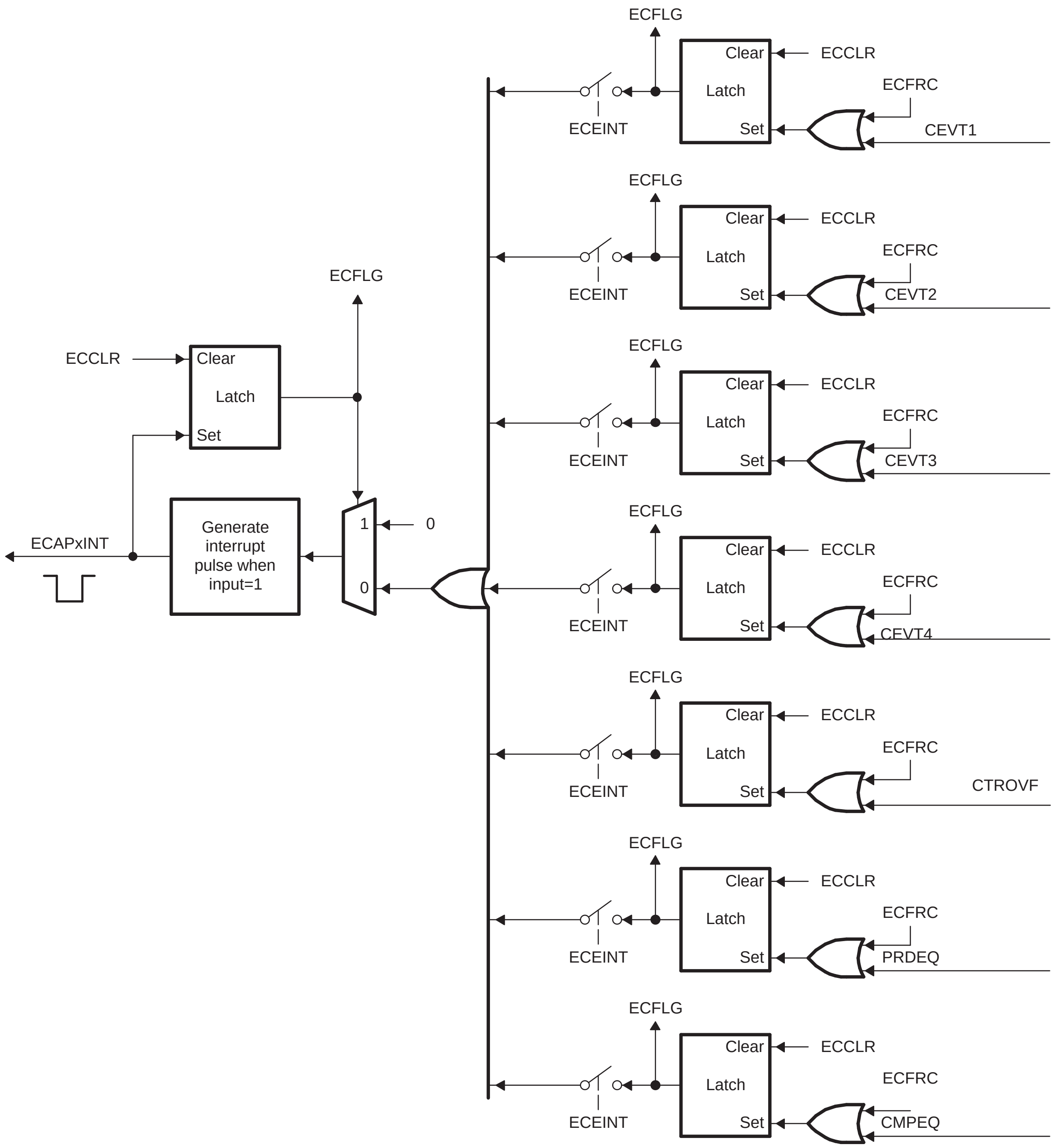
The capture events are edge and sequencer qualified (i.e., ordered in time) by the polarity select and Mod4 gating, respectively.

One of these events can be selected as the interrupt source (from the eCAPx module) going to the PIE.

Seven interrupt events (CEVT1, CEVT2, CEVT3, CEVT4, CNTOVF, CTR=PRD, CTR=CMP) 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 PIE 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: The CEVT1, CEVT2, CEVT3, CEVT4 flags are only active in capture mode (ECCTL2[CAP/APWM == 0]). The CTR=PRD, CTR=CMP flags are only valid in APWM mode (ECCTL2[CAP/APWM == 1]). CNTOVF flag is valid in both modes.

Figure 8. Interrupts in eCAP Module



4.7 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:

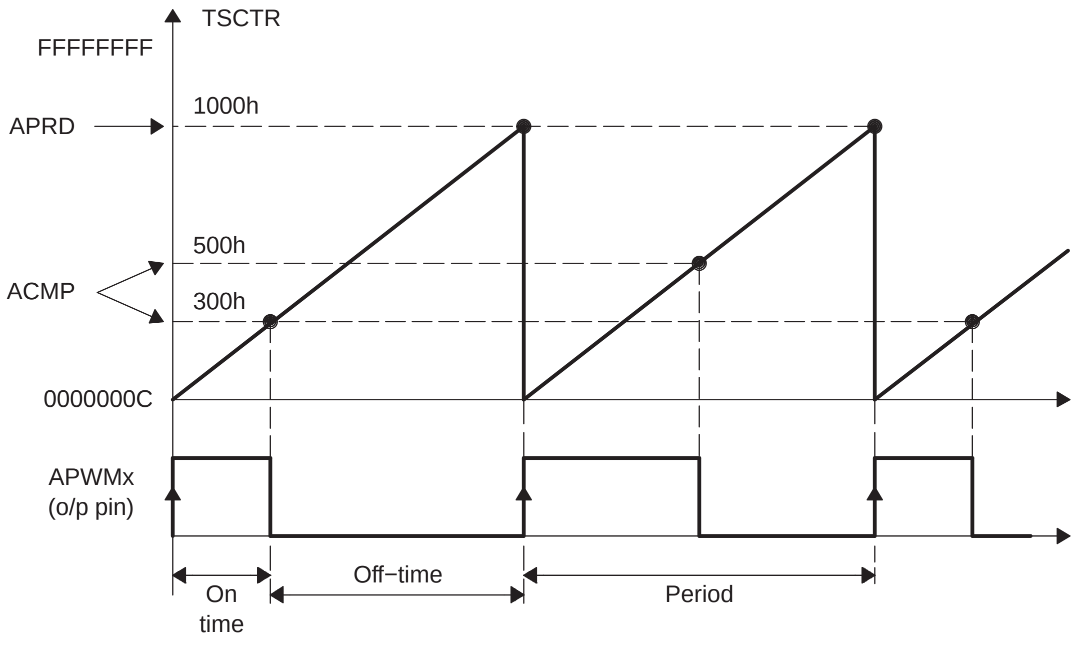
- Immediate - APRD or ACMP are transferred to CAP1 or CAP2 immediately upon writing a new value.
- On period equal, i.e., CTR[31:0] = PRD[31:0]

4.8 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 CTR = PRD 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, i.e., during run-time, you only need to use the shadow registers.

Figure 9. PWM Waveform Details Of APWM Mode Operation



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

```
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 as follows:

```
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
```

5 Capture Module - Control and Status Registers

Figure 10. Time-Stamp Counter Register (TSCTR)

31	0
TSCTR	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 1. Time-Stamp Counter Register (TSCTR) Field Descriptions

Bit(s)	Field	Description
31:0	TSCTR	Active 32-bit counter register that is used as the capture time-base

Figure 11. Counter Phase Control Register (CTRPHS)

31	0
CTRPHS	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 2. Counter Phase Control Register (CTRPHS) Field Descriptions

Bit(s)	Field	Description
31:0	CTRPHS	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.

Figure 12. Capture-1 Register (CAP1)

31	0
CAP1	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 3. Capture-1 Register (CAP1) Field Descriptions

Bit(s)	Field	Description
31:0	CAP1	This register can be loaded (written) by :) Time-Stamp (i.e., counter value TSCTR) during a capture event) Software - may be useful for test purposes / initialization) APRD shadow register (i.e., CAP3) when used in APWM mode

Figure 13. Capture-2 Register (CAP2)

31	0
CAP2	
R/W-0	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 4. Capture-2 Register (CAP2) Field Descriptions

Bit(s)	Field	Description
31:0	CAP2	This register can be loaded (written) by: • Time-Stamp (i.e., counter value) during a capture event • Software - may be useful for test purposes • APRD shadow register (i.e., CAP4) when used in APWM mode

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.

Figure 14. Capture-3 Register (CAP3)

31 … 0
CAP3
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 5. Capture-3 Register (CAP3) Field Descriptions

Bit(s)	Field	Description
31:0	CAP3	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the period shadow (APER) register. You update the PWM period value through this register. In this mode, CAP3 (APRD) shadows CAP1.

Figure 15. Capture-4 Register (CAP4)

31 … 0
CAP4
R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 6. Capture-4 Register (CAP4) Field Descriptions

Bit(s)	Field	Description
31:0	CAP4	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 via this register. In this mode, CAP4 (ACMP) shadows CAP2.

Figure 16. ECAP Control Register 1 (ECCTL1)

15	14	13	12	11	10	9	8
FREE/SOFT		PRESCALE					CAPLDEN
R/W-0		R/W-0					R/W-0

7	6	5	4	3	2	1	0
CTRRST4	CAP4POL	CTRRST3	CAP3POL	CTRRST2	CAP2POL	CTRRST1	CAP1POL
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 7. ECAP Control Register 1 (ECCTL1) Field Descriptions

Bit(s)	Field	Value	Description
15:14	FREE/SOFT		Emulation Control
		00	TSCTR counter stops immediately on emulation suspend

Table 7. ECAP Control Register 1 (ECCTL1) Field Descriptions (continued)

Bit(s)	Field	Value	Description
		01	TSCTR counter runs until = 0
		1x	TSCTR counter is unaffected by emulation suspend (Run Free)
13:9	PRESCALE		Event Filter prescale select
		00000	Divide by 1 (i.e,. no prescale, by-pass the prescaler)
		00001	Divide by 2
		00010	Divide by 4
		00011	Divide by 6
		00100	Divide by 8
		00101	Divide by 10
		...	
		11110	Divide by 60
		11111	Divide by 62
8	CAPLDEN		Enable Loading of CAP1-4 registers on a capture event
		0	Disable CAP1-4 register loads at capture event time.
		1	Enable CAP1-4 register loads at capture event time.
7	CTRRST4		Counter Reset on Capture Event 4
		0	*Do not* reset counter on Capture Event 4 (absolute time stamp operation)
		1	Reset counter after Capture Event 4 time-stamp has been captured (used in difference mode operation)
6	CAP4POL		Capture Event 4 Polarity select
		0	Capture Event 4 triggered on a rising edge (RE)
		1	Capture Event 4 triggered on a falling edge (FE)
5	CTRRST3		Counter Reset on Capture Event 3
		0	*Do not* reset counter on Capture Event 3 (absolute time stamp)
		1	Reset counter after Event 3 time-stamp has been captured (used in difference mode operation)
4	CAP3POL		Capture Event 3 Polarity select
		0	Capture Event 3 triggered on a rising edge (RE)
		1	Capture Event 3 triggered on a falling edge (FE)
3	CTRRST2		Counter Reset on Capture Event 2
		0	*Do not* reset counter on Capture Event 2 (absolute time stamp)
		1	Reset counter after Event 2 time-stamp has been captured (used in difference mode operation)
2	CAP2POL		Capture Event 2 Polarity select
		0	Capture Event 2 triggered on a rising edge (RE)
		1	Capture Event 2 triggered on a falling edge (FE)
1	CTRRST1		Counter Reset on Capture Event 1
		0	*Do not* reset counter on Capture Event 1 (absolute time stamp)
		1	Reset counter after Event 1 time-stamp has been captured (used in difference mode operation)
0	CAP1POL		Capture Event 1 Polarity select
		0	Capture Event 1 triggered on a rising edge (RE)
		1	Capture Event 1 triggered on a falling edge (FE)

Figure 17. ECAP Control Register 2 (ECCTL2)

15–11	10	9	8
Reserved	APWMPOL	CAP/APWM	SWSYNC
R-0	R/W-0	R/W-0	R/W-0

7–6	5	4	3	2–1	0
SYNCO_SEL	SYNCI_EN	TSCTRSTOP	REARM	STOP_WRAP	CONT/ONESHT
R/W-0	R/W-0	R/W-0	R/W-0	R/W-1 R/W-1	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 8. ECAP Control Register 2 (ECCTL2) Field Descriptions

Bit(s)	Field		Description
15:11	Reserved		Reserved
10	APWMPOL		APWM output polarity select. This is applicable only in APWM operating mode
		0	Output is active high (i.e., Compare value defines high time)
		1	Output is active low (i.e., Compare value defines low time)
9	CAP/APWM		CAP/APWM operating mode select
		0	ECAP module operates in capture mode. This mode forces the following configuration: • Inhibits TSCTR resets via CTR = PRD event • Inhibits shadow loads on CAP1 and 2 registers • Permits user to enable CAP1-4 register load • CAPx/APWMx pin operates as a capture input
		1	ECAP module operates in APWM mode. This mode forces the following configuration: • Resets TSCTR on CTR = PRD event (period boundary • Permits shadow loading on CAP1 and 2 registers • Disables loading of time-stamps into CAP1-4 registers • CAPx/APWMx pin operates as a APWM output
8	SWSYNC		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 CTR = PRD event.
		0	Writing a zero has no effect. Reading always returns a zero
		1	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. Note: Selection CTR = PRD is meaningful only in APWM mode; however, you can choose it in CAP mode if you find doing so useful.
7:6	SYNCO_SEL		Sync-Out Select
		00	Select sync-in event to be the sync-out signal (pass through)
		01	Select CTR = PRD event to be the sync-out signal
		10	Disable sync out signal
		11	Disable sync out signal
5	SYNCI_EN		Counter (TSCTR) Sync-In select mode
		0	Disable sync-in option
		1	Enable counter (TSCTR) to be loaded from TSCTR register upon either a SYNCI signal or a S/W force event.
4	TSCTRSTOP		Time Stamp (TSCTR) Counter Stop (freeze) Control
		0	TSCTR stopped
		1	TSCTR free-running
3	RE-ARM		One-Shot Re-Arming Control, i.e. wait for stop trigger. Note: The re-arm function is valid in one shot or continuous mode.
		0	Has no effect (reading always returns a 0)

Table 8. ECAP Control Register 2 (ECCTL2) Field Descriptions (continued)

Bit(s)	Field		Description
		1	Arms the one-shot sequence as follows: 1) Resets the Mod4 counter to zero 2) Unfreezes the Mod4 counter 3) Enables capture register loads
2:1	STOP_WRAP		Stop value for one-shot mode. This is the number (between 1-4) of captures allowed to occur before the CAP(1-4) registers are frozen, i.e., capture sequence is stopped. Wrap value for continuous mode. This is the number (between 1-4) of the capture register in which the circular buffer wraps around and starts again.
		00	Stop after Capture Event 1 in one-shot mode Wrap after Capture Event 1 in continuous mode.
		01	Stop after Capture Event 2 in one-shot mode Wrap after Capture Event 2 in continuous mode.
		10	Stop after Capture Event 3 in one-shot mode Wrap after Capture Event 3 in continuous mode.
		11	Stop after Capture Event 4 in one-shot mode Wrap after Capture Event 4 in continuous mode.
			Notes: STOP_WRAP is compared to Mod4 counter and, when equal, 2 actions occur: • Mod4 counter is stopped (frozen) • Capture register loads are inhibited In one-shot mode, further interrupt events are blocked until re-armed.
0	CONT/ONESHT		Continuous or one-shot mode control (applicable only in capture mode)
		0	Operate in continuous mode
		1	Operate in one-Shot mode

Figure 18. ECAP Interrupt Enable Register (ECEINT)

15–8
Reserved

7	6	5	4	3	2	1	0
CTR=CMP	CTR=CMP	CTROVF	CEVT4	CEVT3	CEVT2	CETV1	Reserved
		R/W	R/W	R/W	R/W	R/W	

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 9. ECAP Interrupt Enable Register (ECEINT) Field Descriptions

Bits	Field	Value	Description
15:8	Reserved		
7	CTR=CMP		Counter Equal Compare Interrupt Enable
		0	Disable Compare Equal as an Interrupt source
		1	Enable Compare Equal as an Interrupt source
6	CTR=CMP		Counter Equal Compare Interrupt Enable
		0	Disable Compare Equal as an Interrupt source
		1	Enable Compare Equal as an Interrupt source
5	CTROVF		Counter Overflow Interrupt Enable
		0	Disabled counter Overflow as an Interrupt source
		1	Enable counter Overflow as an Interrupt source
4	CEVT4		Capture Event 4 Interrupt Enable
		0	Disable Capture Event 1 as an Interrupt source
		1	Capture Event 4 Interrupt Enable
3	CEVT3		Capture Event 3 Interrupt Enable
		0	Disable Capture Event 1 as an Interrupt source
		1	Enable Capture Event 1 as an Interrupt source
2	CEVT2		Capture Event 2 Interrupt Enable
		0	Disable Capture Event 1 as an Interrupt source
		1	Enable Capture Event 1 as an Interrupt source
1	CEVT1		Capture Event 1 Interrupt Enable
		0	Disable Capture Event 1 as an Interrupt source
		1	Enable Capture Event 1 as an Interrupt source
0	Reserved		

The interrupt enable bits (CEVT1, ...) 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/cleared via the ECFRC/ECCLR registers.

The proper procedure for configuring peripheral modes and interrupts is as follows:

- Disable global interrupts
- Stop eCAP counter
- Disable eCAP interrupts
- Configure peripheral registers
- Clear spurious eCAP interrupt flags
- Enable eCAP interrupts
- Start eCAP counter
- Enable global interrupts

Figure 19. ECAP Interrupt Flag Register (ECFLG)

15							8
Reserved							
R-0							

7	6	5	4	3	2	1	0
CTR=CMP	CTR=PRD	CTROVF	CEVT4	CETV3	CEVT2	CETV1	INT
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 10. ECAP Interrupt Flag Register (ECFLG) Field Descriptions

Bits	Field	Value	Description[(1)]
15:8	Reserved		
7	CTR=CMP		Compare Equal Compare Status Flag. This flag is active only in APWM mode.
		0	Indicates no event occurred
		1	Indicates the counter (TSCTR) reached the compare register value (ACMP)
6	CTR=PRD		Counter Equal Period Status Flag. This flag is only active in APWM mode.
		0	Indicates no event occurred
		1	Indicates the counter (TSCTR) reached the period register value (APER) and was reset.
5	CTROVF		Counter Overflow Status Flag. This flag is active in CAP and APWM mode.
		0	Indicates no event occurred.
		1	Indicates the counter (TSCTR) has made the transition from FFFFFFFF " 00000000
4	CEVT4		Capture Event 4 Status Flag This flag is only active in CAP mode.
		0	Indicates no event occurred
		1	Indicates the fourth event occurred at ECAPx pin
3	CEVT3		Capture Event 3 Status Flag. This flag is active only in CAP mode.
		0	Indicates no event occurred.
		1	Indicates the third event occurred at ECAPx pin.
2	CEVT2		Capture Event 2 Status Flag. This flag is only active in CAP mode.
		0	Indicates no event occurred.
		1	Indicates the second event occurred at ECAPx pin.
1	CEVT1		Capture Event 1 Status Flag. This flag is only active in CAP mode.
		0	Indicates no event occurred.
		1	Indicates the first event occurred at ECAPx pin.
0	INT		Global Interrupt Status Flag
		0	Indicates no interrupt generated.
		1	Indicates that an interrupt was generated.

(1) If the event interrupts are enabled (via the ECEINT register) and any one of the above event flags is set to 1 and the INT flag is 0, then an interrupt pulse is generated and the INT flag is set to 1. No further interrupt pulses are generated. Further interrupt pulses are only generated if the INT flag bit is cleared (via the ECCLR register) and one or more event flag bits is set to 1. If all event flag bits are manually cleared, then no further pulses are generated.

Figure 20. ECAP Interrupt Clear Register (ECCLR)

15							8
Reserved							
R-0							

7	6	5	4	3	2	1	0
CTR=CMP	CTR=PRD	CTROVF	CEVT4	CETV3	CETV2	CETV1	INT
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 11. ECAP Interrupt Clear Register (ECCLR) Field Descriptions

Bits	Field		Description
15:8	Reserved		
7	CTR=CMP		Counter Equal Compare Status Flag
		0	Writing a 0 has no effect. Always reads back a 0
		1	Writing a 1 clears the CTR=CMP flag condition
6	CTR=PRD		Counter Equal Period Status Flag
		0	Writing a 0 has no effect. Always reads back a 0
		1	Writing a 1 clears the CTR=PRD flag condition
5	CTROVF		Counter Overflow Status Flag
		0	Writing a 0 has no effect. Always reads back a 0
		1	Writing a 1 clears the CTROVF flag condition
4	CEVT4		Capture Event 4 Status Flag
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CEVT3 flag condition.
3	CEVT3		Capture Event 3 Status Flag
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CEVT3 flag condition.
2	CEVT2		Capture Event 2 Status Flag
		0	Writing a 0 has no effect. Always reads back a 0.
		0	Writing a 1 clears the CEVT2 flag condition.
1	CEVT1		Capture Event 1 Status Flag
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the CEVT1 flag condition.
0	INT		Global Interrupt Clear Flag
		0	Writing a 0 has no effect. Always reads back a 0.
		1	Writing a 1 clears the INT flag and enable further interrupts to be generated if any of the event flags are set to 1.

Figure 21. ECAP Interrupt Forcing Register (ECFRC)

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
CTR=CMP	CTR=PRD	CTROVF	CEVT4	CETV3	CETV2	CETV1	reserved
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 12. ECAP Interrupt Forcing Register (ECFRC) Field Descriptions

Bits	Field	Value	Description[(1)]
15:8	Reserved	0	
7	CTR-CMP		Force Counter Equal Compare Interrupt
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CTR=CMP flag bit.

(1) Writing a 1 in any of the above bits causes the respective flag bit to be set to 1. An interrupt pulse is only generated to the PIE if the INT flag bit is 0 before the write. As soon as the interrupt pulse is generated, the INT flag bit is set to 1. If the INT flag bit was already 1 before the operation, then no interrupt pulse is generated. A pulse is generated if the INT flag bit is cleared (via the ECCLR register) and any of the other flag bits are 1.

Table 12. ECAP Interrupt Forcing Register (ECFRC) Field Descriptions (continued)

Bits	Field	Value	Description[(1)]
6	CTR=PRD		Force Counter Equal Period Interrupt
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CTR=PRD flag bit.
5	CTROVF		Force Counter Overflow
		0	No effect. Always reads back a 0.
		1	Writing a 1 to this bit sets the CTROVF flag bit.
4	CEVT4		Force Capture Event 4
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CEVT4 flag bit
3	CEVT3		Force Capture Event 3
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CEVT3 flag bit
2	CEVT2		Force Capture Event 2
		0	No effect. Always reads back a 0.
		1	Writing a 1 sets the CEVT2 flag bit.
1	CEVT1		Force Capture Event 1
		0	No effect. Always reads back a 0.
		0	Sets the CEVT1 flag bit.
0	reserved	0	

6 Register Mapping

Table 13 shows the eCAP module control and status register set.

Table 13. Control and Status Register Set

Name	Offset	Size (x16)[(1)]	Description
Time Base Module Registers			
TSCTR	0x0000	2	Time-Stamp Counter
CTRPHS	0x0002	2	Counter Phase Offset Value Register
CAP1	0x0004	2	Capture 1 Register
CAP2	0x0006	2	Capture 2 Register
CAP3	0x0008	2	Capture 3 Register
CAP4	0x000A	2	Capture 4 Register
reserved	0x000C - 0x0013	8	
ECCTL1	0x0014	1	Capture Control Register 1
ECCTL2	0x0015	1	Capture Control Register 2
ECEINT	0x0016	1	Capture Interrupt Enable Register
ECFLG	0x0017	1	Capture Interrupt Flag Register
ECCLR	0x0018	1	Capture Interrupt Clear Register
ECFRC	0x0019	1	Capture Interrupt Force Register
Reserved	0x001A - 0x001F	6	

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

7 Application of the ECAP Module

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, the examples use the eCAP “C” header files. Below are useful #defines which will help in the understanding of the examples.

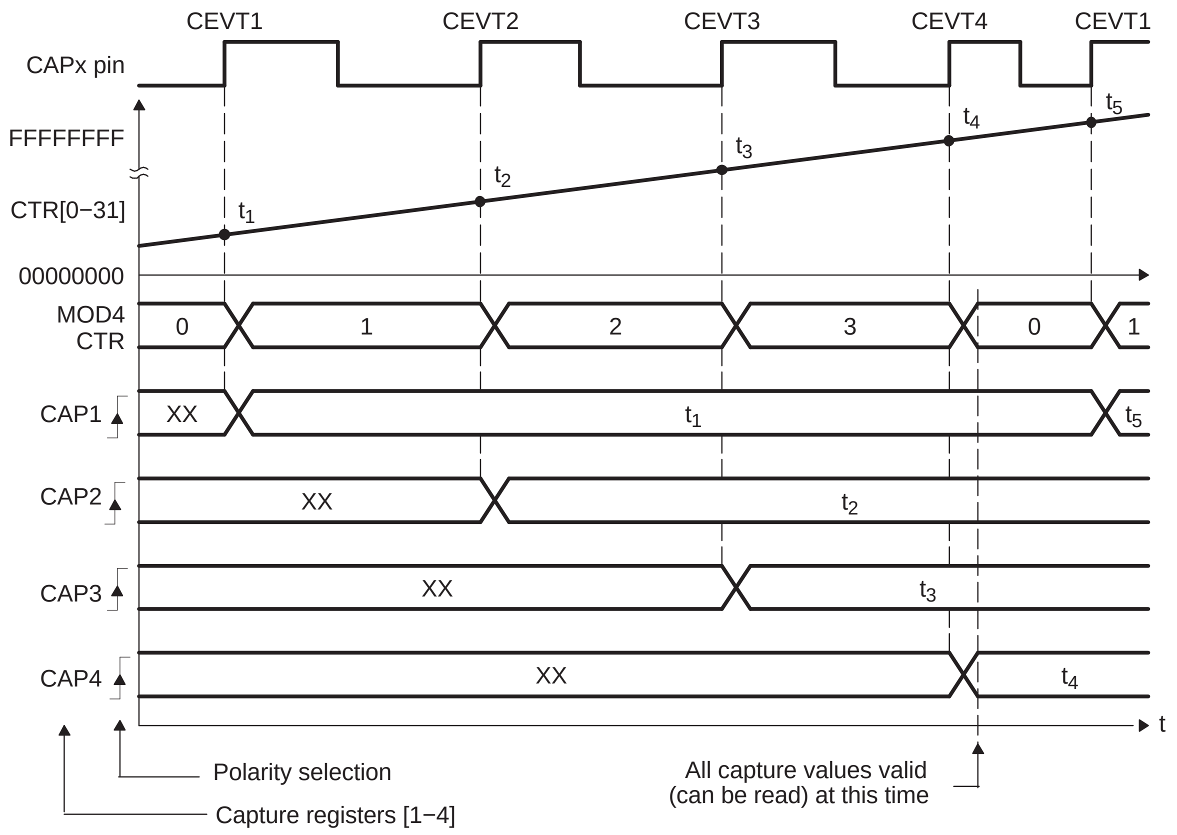
```
// ECCTL1 ( ECAP Control Reg 1)
//==========================
// CAPxPOL bits
#define    EC_RISING              0x0
#define    EC_FALLING             0x1

// 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
```

7.1 Example 1 - Absolute Time-Stamp Operation Rising Edge Trigger

Figure 22 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 (i.e., time-stamp) is first captured, then Mod4 counter is incremented to the next state. When the TSCTR reaches FFFFFFFF (i.e. maximum value), it wraps around to 00000000 (not shown in Figure 22), if this occurs, the CTROVF (counter overflow) flag is set, and an interrupt (if enabled) occurs, CTROVF (counter overflow) Flag is set, and an Interrupt (if enabled) occurs. Captured Time-stamps are valid at the point indicated by the diagram, i.e. after the 4th event, hence event CEVT4 can conveniently be used to trigger an interrupt and the CPU can read data from the CAPx registers.

Figure 22. Capture Sequence for Absolute Time-stamp and Rising Edge Detect



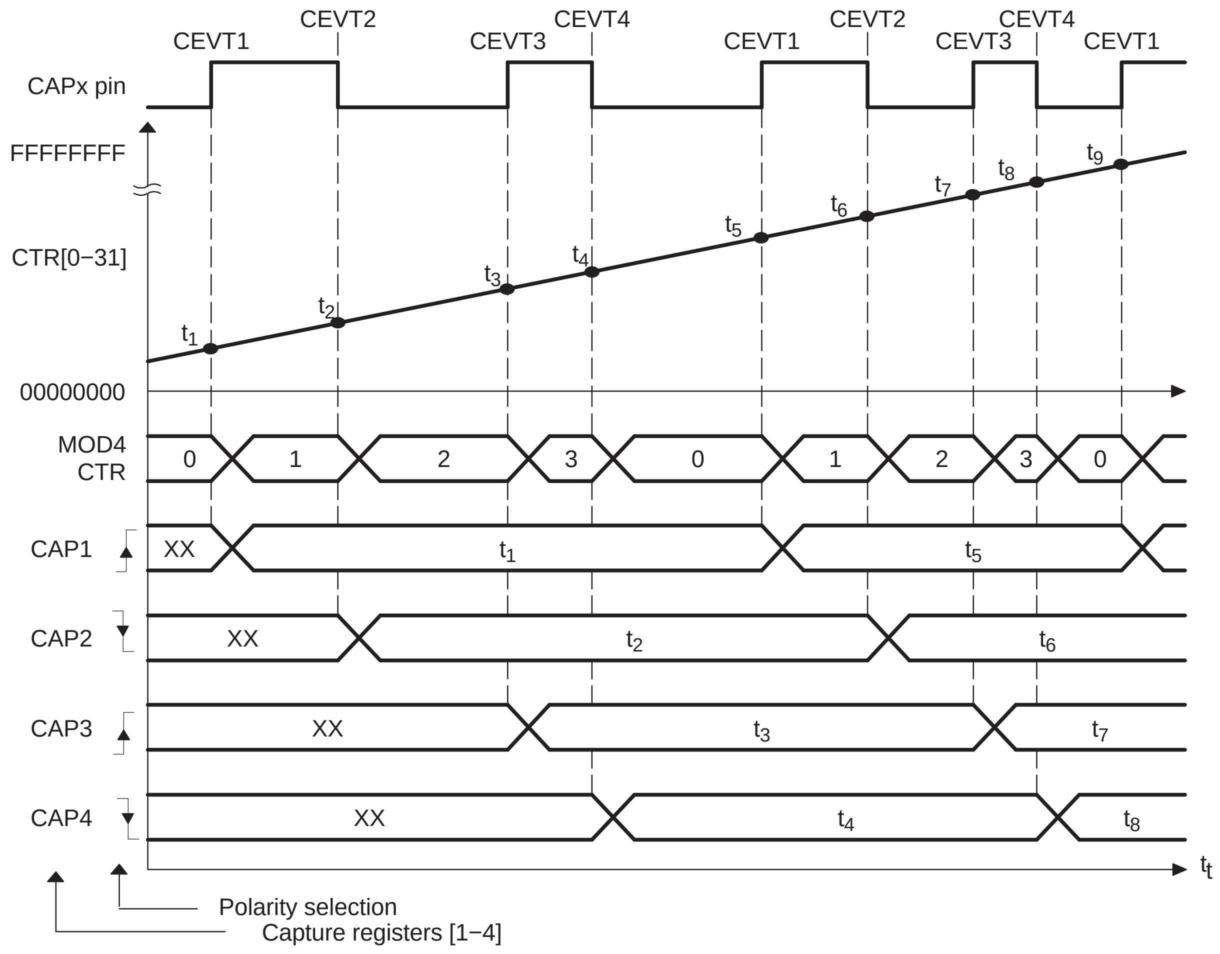
7.1.1 Code snippet for CAP mode Absolute Time, Rising Edge Trigger

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

// Initialization Time
//=======================
// ECAP module 1 config
    ECap1Regs.ECCTL1.bit.CAP1POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CAP2POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CAP3POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CAP4POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CTRRST1 = EC_ABS_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST2 = EC_ABS_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST3 = EC_ABS_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST4 = EC_ABS_MODE;
    ECap1Regs.ECCTL1.bit.CAPLDEN = EC_ENABLE;
    ECap1Regs.ECCTL1.bit.PRESCALE = EC_DIV1;

    ECap1Regs.ECCTL2.bit.CAP_APWM = EC_CAP_MODE;
    ECap1Regs.ECCTL2.bit.CONT_ONESHT = EC_CONTINUOUS;
    ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS;
    ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE;
    ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;        // Allow TSCTR to run


// Run Time ( e.g. CEVT4 triggered ISR call)
//==========================================
    TSt1 = ECap1Regs.CAP1;        // Fetch Time-Stamp captured at t1
    TSt2 = ECap1Regs.CAP2;        // Fetch Time-Stamp captured at t2
    TSt3 = ECap1Regs.CAP3;        // Fetch Time-Stamp captured at t3
    TSt4 = ECap1Regs.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
```

7.2 Example 2 - Absolute Time-Stamp Operation Rising and Falling Edge Trigger

In Figure 23 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, i.e: Period1 = $t_3 - t_1$, Period2 = $t_5 - t_3$, …etc. Duty Cycle1 (on-time %) = $(t_2 - t_1)$ / Period1 x 100%, etc. Duty Cycle1 (off-time %) = $(t_3 - t_2)$ / Period1 x 100%, etc.

Figure 23. Capture Sequence for Absolute Time-stamp With Rising and Falling Edge Detect



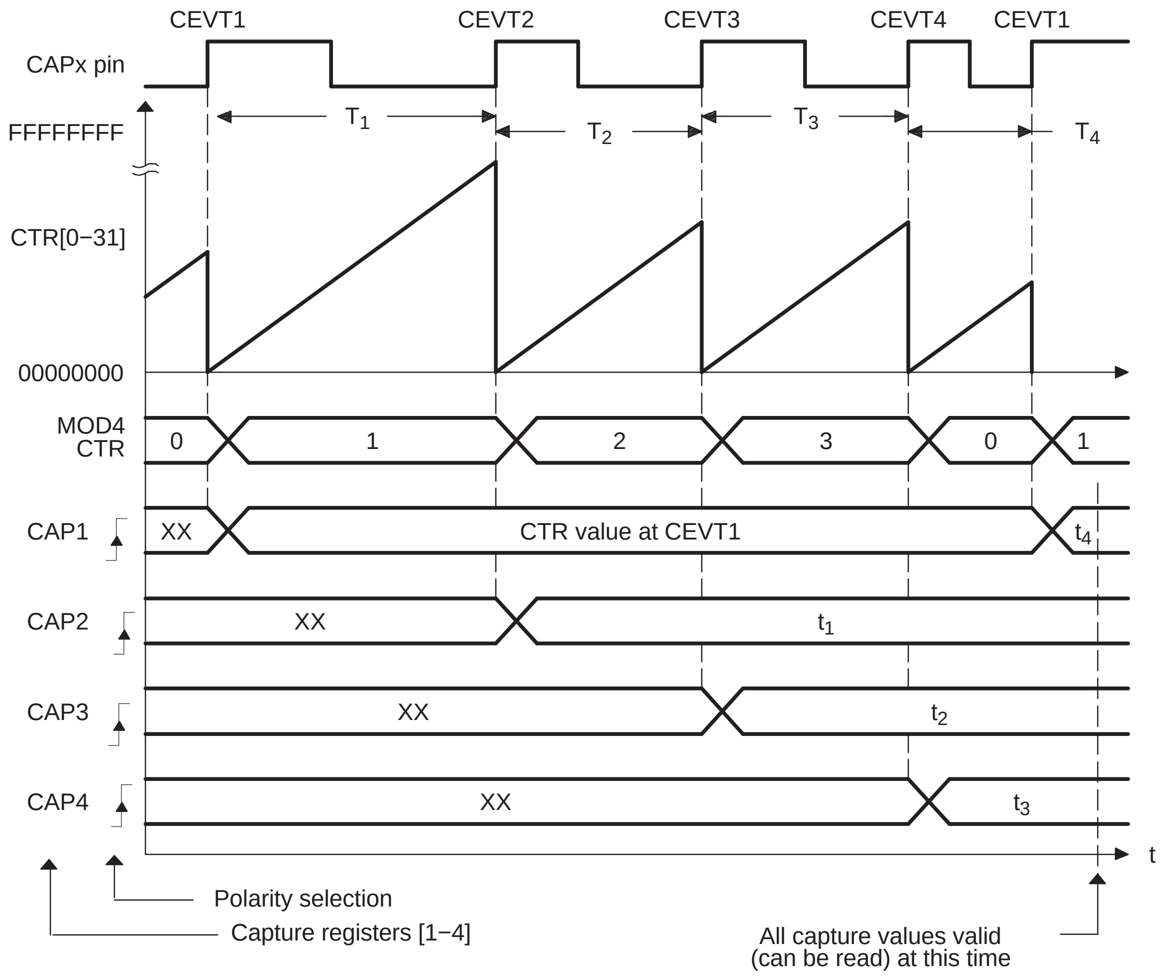
7.2.1 Code snippet for CAP mode Absolute Time, Rising & Falling Edge Triggers

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

// Initialization Time
//=======================
// ECAP module 1 config
    ECap1Regs.ECCTL1.bit.CAP1POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CAP2POL = EC_FALLING;
    ECap1Regs.ECCTL1.bit.CAP3POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CAP4POL = EC_FALLING;
    ECap1Regs.ECCTL1.bit.CTRRST1 = EC_ABS_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST2 = EC_ABS_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST3 = EC_ABS_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST4 = EC_ABS_MODE;
    ECap1Regs.ECCTL1.bit.CAPLDEN = EC_ENABLE;
    ECap1Regs.ECCTL1.bit.PRESCALE = EC_DIV1;

    ECap1Regs.ECCTL2.bit.CAP_APWM = EC_CAP_MODE;
    ECap1Regs.ECCTL2.bit.CONT_ONESHT = EC_CONTINUOUS;
    ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS;
    ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE;
    ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;        // Allow TSCTR to run


// Run Time ( e.g. CEVT4 triggered ISR call)
//==========================================
    TSt1 = ECap1Regs.CAP1;        // Fetch Time-Stamp captured at t1
    TSt2 = ECap1Regs.CAP2;        // Fetch Time-Stamp captured at t2
    TSt3 = ECap1Regs.CAP3;        // Fetch Time-Stamp captured at t3
    TSt4 = ECap1Regs.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
```

7.3 Example 3 - Time Difference (Delta) Operation Rising Edge Trigger

This example Figure 24 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 (i.e. Time-Stamp) is captured first, and then TSCTR is reset to Zero. The Mod4 counter then increments to the next state. If TSCTR reaches FFFFFFFF (i.e. Max value), before the next event, it wraps around to 00000000 and continues, a CNTOVF (counter overflow) Flag is set, and an Interrupt (if enabled) occurs. The advantage of Delta-time Mode is that the CAPx contents directly give timing data without the need for CPU calculations, i.e. Period1 = T_1, Period2 = T_2,…etc. As shown in the diagram, 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 24. Capture Sequence for Delta Mode Time-stamp and Rising Edge Detect



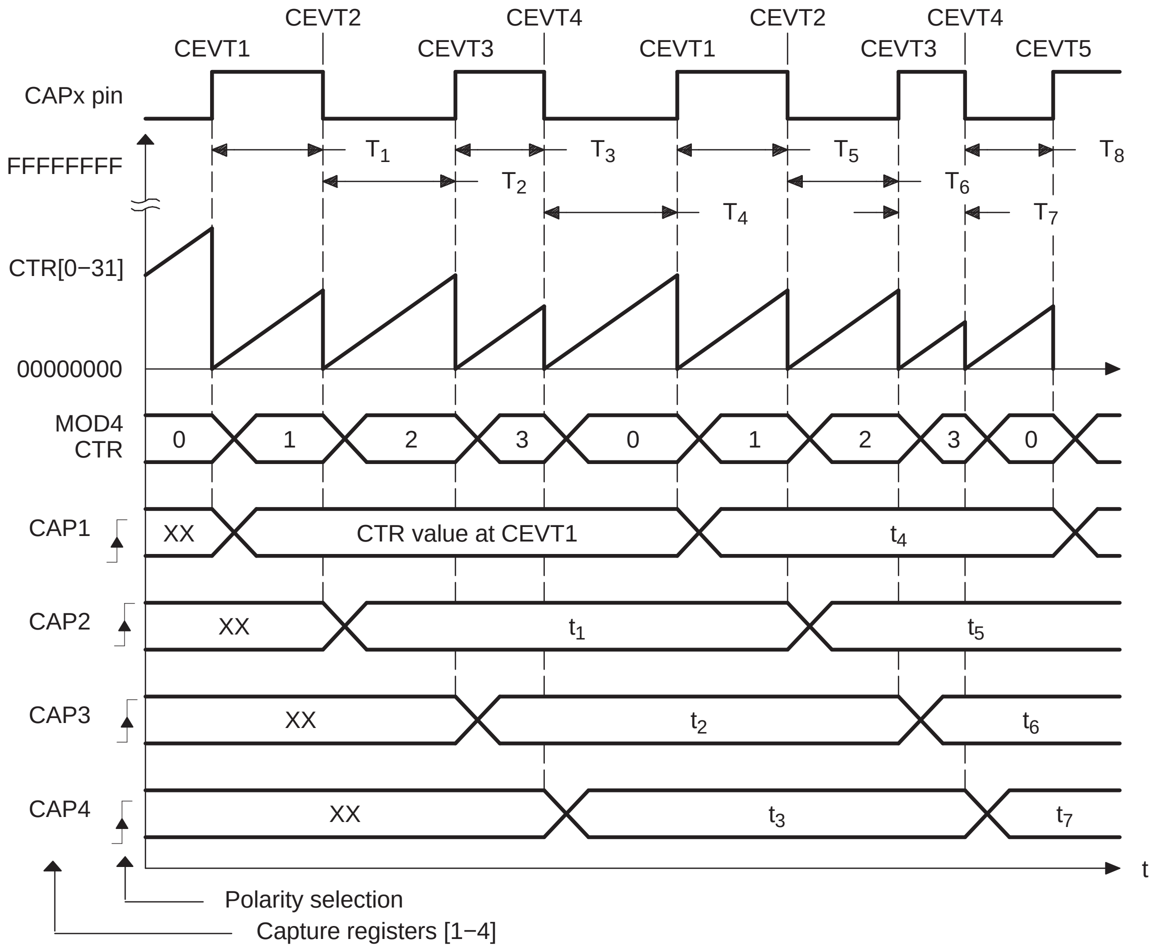
7.3.1 Code snippet for CAP mode Delta Time, Rising Edge Trigger

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

// Initialization Time
//=======================
// ECAP module 1 config
    ECap1Regs.ECCTL1.bit.CAP1POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CAP2POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CAP3POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CAP4POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CTRRST1 = EC_DELTA_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST2 = EC_DELTA_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST3 = EC_DELTA_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST4 = EC_DELTA_MODE;
    ECap1Regs.ECCTL1.bit.CAPLDEN = EC_ENABLE;
    ECap1Regs.ECCTL1.bit.PRESCALE = EC_DIV1;

    ECap1Regs.ECCTL2.bit.CAP_APWM = EC_CAP_MODE;
    ECap1Regs.ECCTL2.bit.CONT_ONESHT = EC_CONTINUOUS;
    ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS;
    ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE;
    ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;        // Allow TSCTR to run


// Run Time ( e.g. CEVT1 triggered ISR call)
//==========================================
// Note: here Time-stamp directly represents the Period value.
    Period4 = ECap1Regs.CAP1;    // Fetch Time-Stamp captured at T1
    Period1 = ECap1Regs.CAP2;    // Fetch Time-Stamp captured at T2
    Period2 = ECap1Regs.CAP3;    // Fetch Time-Stamp captured at T3
    Period3 = ECap1Regs.CAP4;    // Fetch Time-Stamp captured at T4
```

7.4 Example 4 - Time Difference (Delta) Operation Rising and Falling Edge Trigger

In Figure 25 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, i.e: Period1 = T_1+T_2, Period2 = T_3+T_4, …etc Duty Cycle1 (on-time %) = T_1 / Period1 x 100%, etc Duty Cycle1 (off-time %) = T_2 / Period1 x 100%, etc

Figure 25. Capture Sequence for Delta Mode Time-stamp With Rising and Falling Edge Detect



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, i.e. during run-time, only the shadow registers must be used.

7.4.1 Code snippet for CAP mode Delta Time, Rising and Falling Edge Triggers

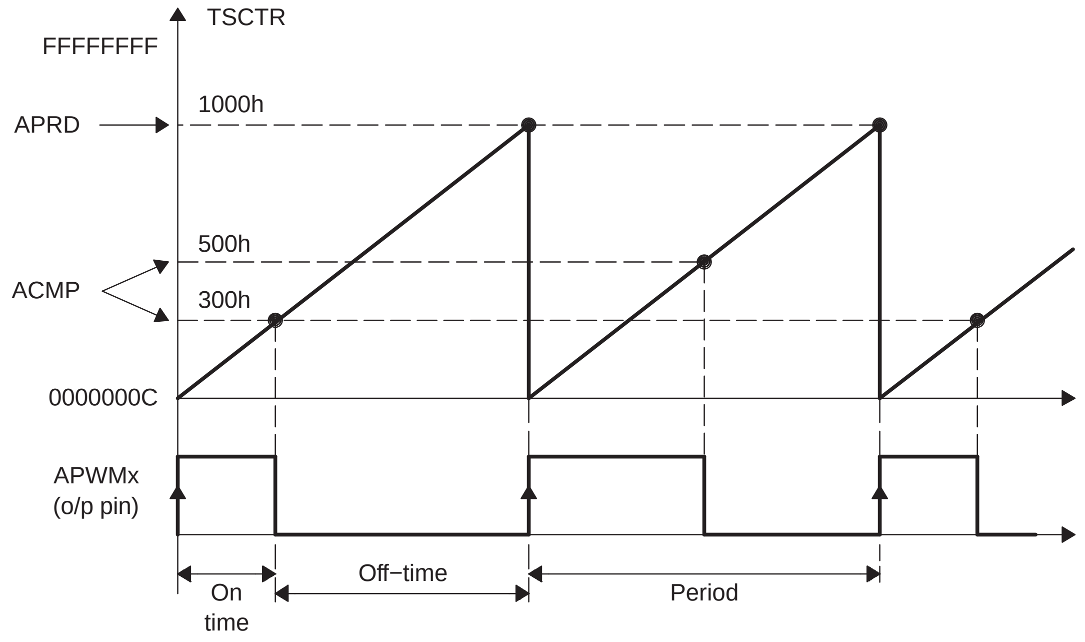
```
// Code snippet for CAP mode Delta Time, Rising and Falling
       // edge triggers

// Initialization Time
//=======================
// ECAP module 1 config
    ECap1Regs.ECCTL1.bit.CAP1POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CAP2POL = EC_FALLING;
    ECap1Regs.ECCTL1.bit.CAP3POL = EC_RISING;
    ECap1Regs.ECCTL1.bit.CAP4POL = EC_FALLING;
    ECap1Regs.ECCTL1.bit.CTRRST1 = EC_DELTA_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST2 = EC_DELTA_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST3 = EC_DELTA_MODE;
    ECap1Regs.ECCTL1.bit.CTRRST4 = EC_DELTA_MODE;
    ECap1Regs.ECCTL1.bit.CAPLDEN = EC_ENABLE;
    ECap1Regs.ECCTL1.bit.PRESCALE = EC_DIV1;

    ECap1Regs.ECCTL2.bit.CAP_APWM = EC_CAP_MODE;
    ECap1Regs.ECCTL2.bit.CONT_ONESHT = EC_CONTINUOUS;
    ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS;
    ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE;
    ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;        // Allow TSCTR to run


// Run Time ( e.g. CEVT1 triggered ISR call)
//==========================================
// Note: here Time-stamp directly represents the Duty cycle values.
    DutyOnTime1 =  ECap1Regs.CAP2;    // Fetch Time-Stamp captured at T2
    DutyOffTime1 = ECap1Regs.CAP3;    // Fetch Time-Stamp captured at T3
    DutyOnTime2 =  ECap1Regs.CAP4;    // Fetch Time-Stamp captured at T4
    DutyOffTime2 = ECap1Regs.CAP1;    // Fetch Time-Stamp captured at T1

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

8 Application of the APWM Mode

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 APWMx. 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. Note here values are in hexadecimal ("h") notation.

8.1 Example 1 - Simple PWM Generation (Independent Channel/s)

Figure 26. PWM Waveform Details of APWM Mode Operation



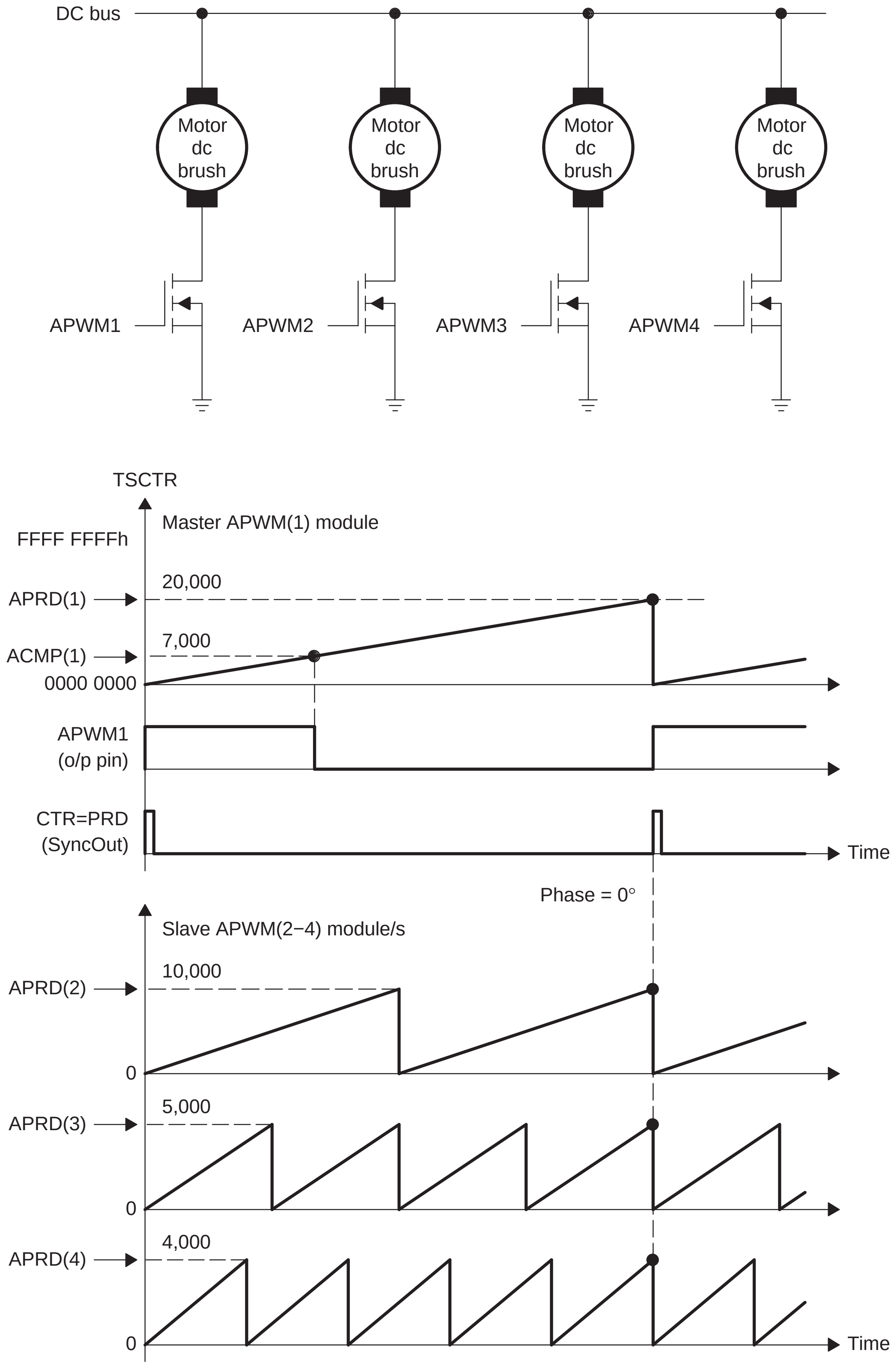
Example 1. Code Snippet for APWM Mode

```
// Code snippet for APWM mode Example 1

// Initialization Time
//=======================
// ECAP module 1 config
   ECap1Regs.CAP1 = 0x1000;                      // Set period value
   ECap1Regs.CTRPHS = 0x0;                       // make phase zero
   ECap1Regs.ECCTL2.bit.CAP_APWM = EC_APWM_MODE;
   ECap1Regs.ECCTL2.bit.APWMPOL = EC_ACTV_HI;    // Active high
   ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE;   // Synch not used
   ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS; // Synch not used
   ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;      // Allow TSCTR to run

// Run Time (Instant 1, e.g. ISR call)
//======================
   ECap1Regs.CAP2 = 0x300;      // Set Duty cycle i.e. compare value

// Run Time (Instant 2, e.g. another ISR call)
//======================
   ECap1Regs.CAP2 = 0x500;      // Set Duty cycle i.e. compare value
```

8.2 Example 2 - Multi-channel PWM Generation With Synchronization

This example 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 (F1 = 1/20,000) requirement. Here Slave2 Freq = 2 x F1, Slave3 Freq = 4 x F1 and Slave4 Freq = 5 x F1. Note here values are in decimal notation. Also only the APWM1 output waveform is shown.

Figure 27. Multichannel PWM Example Using 4 eCAP Modules



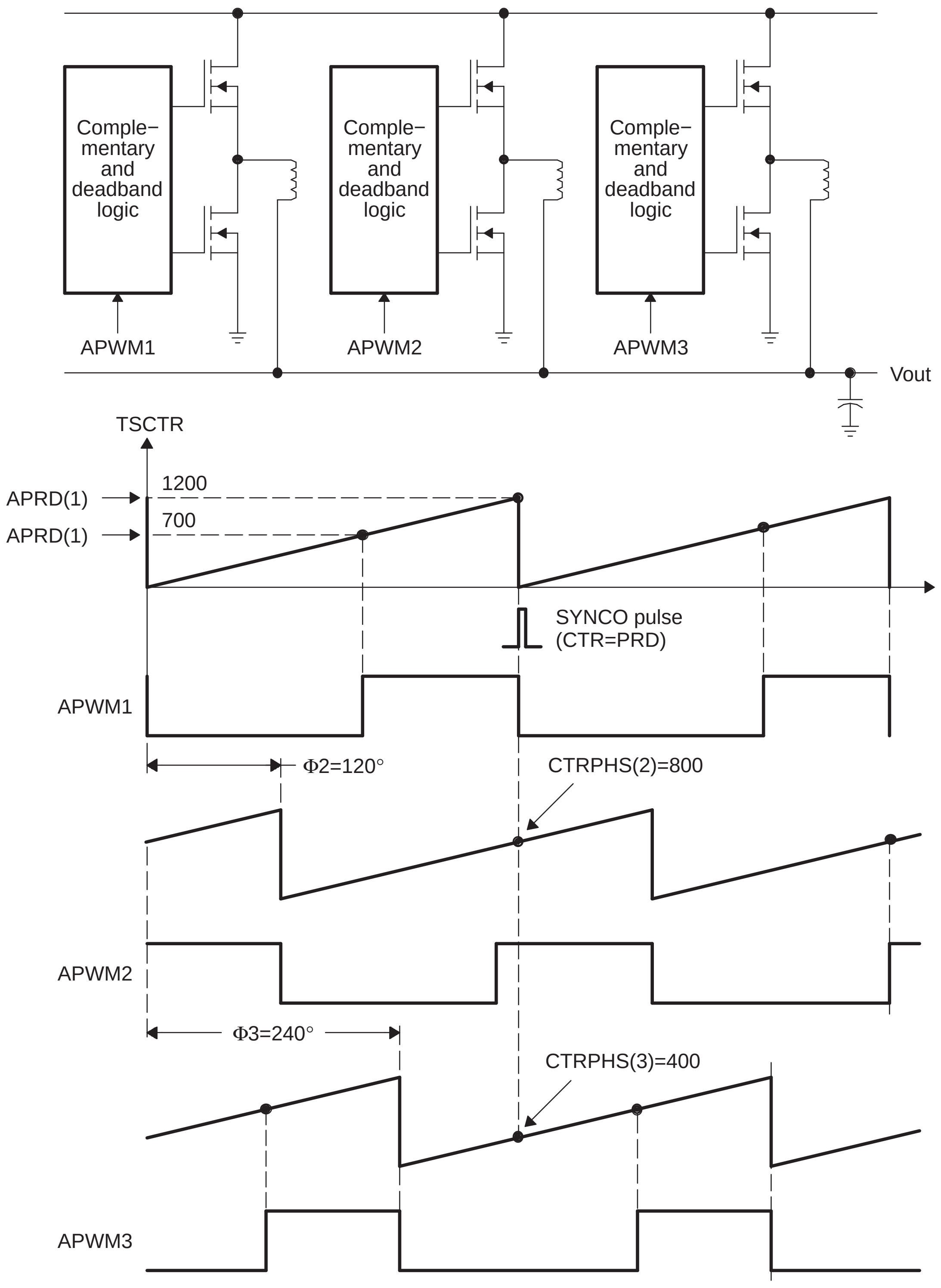
Example 2. Code Snippet for APWM Mode

```
// Code snippet for APWM mode Example 2

// Initialization Time
//=======================
// ECAP module 1 config
   ECap1Regs.CAP1 = 20000;                        // Set period value
   ECap1Regs.CTRPHS = 0;                          // make phase zero
   ECap1Regs.ECCTL2.bit.CAP_APWM = EC_APWM_MODE;
   ECap1Regs.ECCTL2.bit.APWMPOL = EC_ACTV_HI;
   ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE;    // No sync in for Master
   ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_CTR_PRD;   // eCAP1 is Master
   ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;       // Allow TSCTR to run

// ECAP module 2 config
   ECap2Regs.CAP1 = 10000;                        // Set period value
   ECap2Regs.CTRPHS = 0;
   ECap2Regs.ECCTL2.bit.CAP_APWM = EC_APWM_MODE;
   ECap2Regs.ECCTL2.bit.APWMPOL = EC_ACTV_HI;
   ECap2Regs.ECCTL2.bit.SYNCI_EN = EC_ENABLE;     // slaved off master
   ECap2Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCI;     // sync "flow-through"
   ECap2Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;       // Allow TSCTR to run

// ECAP module 3 config
   ECap3Regs.CAP1 = 5000;                         // Set period value
   ECap3Regs.CTRPHS = 0;
   ECap3Regs.ECCTL2.bit.CAP_APWM = EC_APWM_MODE;
   ECap3Regs.ECCTL2.bit.APWMPOL = EC_ACTV_HI;
   ECap3Regs.ECCTL2.bit.SYNCI_EN = EC_ENABLE;     // slaved off master
   ECap3Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCI;     // sync "flow-through"
   ECap3Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;       // Allow TSCTR to run


// ECAP module 4 config
   ECap4Regs.CAP1 = 4000;                         // Set period value
   ECap4Regs.CTRPHS = 0;
   ECap4Regs.ECCTL2.bit.CAP_APWM = EC_APWM_MODE;
   ECap4Regs.ECCTL2.bit.APWMPOL = EC_ACTV_HI;
   ECap4Regs.ECCTL2.bit.SYNCI_EN = EC_ENABLE;     // slaved off master
   ECap4Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS; // "break the chain"
   ECap4Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;       // Allow TSCTR to run


// 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
```

8.3 Example 3 - Multi-channel PWM Generation With Phase Control

In this example 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), i.e. 0°, then Leg 2 need 120° off-set and Leg 3 needs 240° off-set. The waveforms in Figure 28 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 28. Multi-phase (channel) Interleaved PWM Example Using 3 eCAP Modules



Example 3. Code Snippet for APWM Mode

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

// Initialization Time
//=======================
// ECAP module 1 config
  ECap1Regs.CAP1 = 1200;                         // Set period value
  ECap1Regs.CTRPHS = 0;                          // make eCAP1 reference phase = zero
  ECap1Regs.ECCTL2.bit.CAP_APWM = EC_APWM_MODE;
  ECap1Regs.ECCTL2.bit.APWMPOL = EC_ACTV_HI;
  ECap1Regs.ECCTL2.bit.SYNCI_EN = EC_DISABLE;    // No sync in for Master
  ECap1Regs.ECCTL2.bit.SYNCO_SEL = EC_CTR_PRD;   // eCAP1 is Master
  ECap1Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;       // Allow TSCTR to run

// ECAP module 2 config
  ECap2Regs.CAP1 = 1200;                         // Set period value
  ECap2Regs.CTRPHS = 800;                        // Phase offset = 1200-400 = 120 deg
  ECap2Regs.ECCTL2.bit.CAP_APWM = EC_APWM_MODE;
  ECap2Regs.ECCTL2.bit.APWMPOL = EC_ACTV_HI;
  ECap2Regs.ECCTL2.bit.SYNCI_EN = EC_ENABLE;     // slaved off master
  ECap2Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCI;     // sync "flow-through"
  ECap2Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;       // Allow TSCTR to run

// ECAP module 3 config
  ECap3Regs.CAP1 = 1200;                         // Set period value
  ECap3Regs.CTRPHS = 400;                        // Phase offset = 1200-800 = 240 deg
  ECap3Regs.ECCTL2.bit.CAP_APWM = EC_APWM_MODE;
  ECap3Regs.ECCTL2.bit.APWMPOL = EC_ACTV_HI;
  ECap3Regs.ECCTL2.bit.SYNCI_EN = EC_ENABLE;     // slaved off master
  ECap3Regs.ECCTL2.bit.SYNCO_SEL = EC_SYNCO_DIS; // "break the chain"
  ECap3Regs.ECCTL2.bit.TSCTRSTOP = EC_RUN;       // Allow TSCTR to run



// 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
```

Appendix A Revision History

Technical changes were made to this document to make it a SPRU807A from SPRU807.

Table A-1. Changes Made in This Revision

Location	Additions, Deletions, Changes
Figure 3	Modified the Capture Function Diagram by changing STOPVALUE to STOP_WRAP
Figure 4	Moved Event Prescale Control figure from Section 5 to Section 4.1
Figure 5	Moved Prescale Function Waveforms figure from Section 5 to Section 4.1
Figure 8	Moved Interrupts in eCAP Module figure from Section 5 to Section 4.6
Figure 6	Added Details of the Continuous/One-shot Block figure
Figure 7	Added Details of the Counter and Synchronization Block figure
Section 4.8	Added third bullet to APWM Mode Operation on initialization
Figure 9	Added PWM Waveform Details Of APWM Mode Operation figure
Table 2	Changed description of the CTRPHS bits in the Counter Phase Control Register (CTRPHS)
Table 4	Moved the note following the Capture-4 Register (CAP4) Field Descriptions to follow the Capture-2 Register (CAP2) Field Descriptions
Figure 17	Modified the ECAP Control Register 2 (ECCTL2) by changing bits 2:1 from STOPVALUE to STOP_WRAP
Section 8	Added Application of the APWM Mode section

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
Low Power Wireless	www.ti.com/lpw	Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless