

UCD30xx

Timer Modules

Programmer's Manual

Literature Number: xxxxxx

Date

Table of Contents

UCD30xx	1
Timer Modules	1
Programmer's Manual	1
Literature Number: xxxxxx.....	1
Date	1
1 Timer Module Overview.....	3
2 T24 – 24 Bit Free-Running Timer with Capture and Compare	3
2.1 T24 Clock Source, Prescaler and Counter	3
2.2 T24 Capture Blocks	4
2.3 T24 Compare Blocks	5
2.4 T24 Interrupts.....	6
3 T16PWMx - 16 Bit PWM Timers	6
3.1 T16PWMx Summary	7
3.2 T16PWMx Prescaler and Counter	7
3.3 T16PWMx Compare Blocks.....	7
3.4 T16 Shadow Bit.	8
3.5 T16 Interrupts.....	9
3.6 Using the T16 for a Timer Interrupt	9
3.7 Using the T16 for PWM generation.....	9
4 WD – Watchdog	10
4.1 Watchdog prescale and counter.	10
4.2 Watchdog compare blocks.....	10
4.3 Watchdog Protect Bit.....	10
4.4 Watchdog Timer Example	11
4.5 Warnings for Watchdog Status Register.....	11
4.6 Metastable states in the Watchdog Logic	12
4.7 System Fault Recovery Basics.....	12
5 Timer Module Register Reference.....	12
5.1 24-bit Counter Data Register (T24CNTDAT).....	13
5.2 24-bit Counter Control Register (T24CNTCTRL)	13
5.3 24-bit Capture Channel 0-1 Data Register (T24CAPxDAT).....	13
5.4 24-bit Capture Channel 0-1 Control Register (T24CAPxCTRL).....	14
5.5 24-bit Capture I/O Control and Data Register (T24CAPIO)	15
5.6 24-bit Output Compare Channel 0-1 Data Register (T24CMPxDAT).....	16
5.7 24-bit Output Compare Channel 0 Control Register (T24CMP0CTRL).....	16
5.8 24-bit Output Compare Channel 1 Control Register (T24CMP1CTRL).....	16
5.9 PWMx Counter Data Register (T16PWMxCNTDAT)	17
5.10 PWMx Counter Control Register (T16PWMxCNTCTRL).....	18
5.11 PWMx 16-bit Compare Channel 0-1 Data Register (T16PWMxCMPyDAT).....	19
5.12 PWMx Compare Control Register (T16PWMxCMPCTRL)	19
5.13 Watchdog Status (WDST)	21
5.14 Watchdog Control (WDCTRL)	21
6 Other Peripheral Registers Mentioned.....	23
6.1 System Exception Status Register (SYSESR)	23

Timer Module Overview

The Timer Module contains a total of 6 timers. They are:

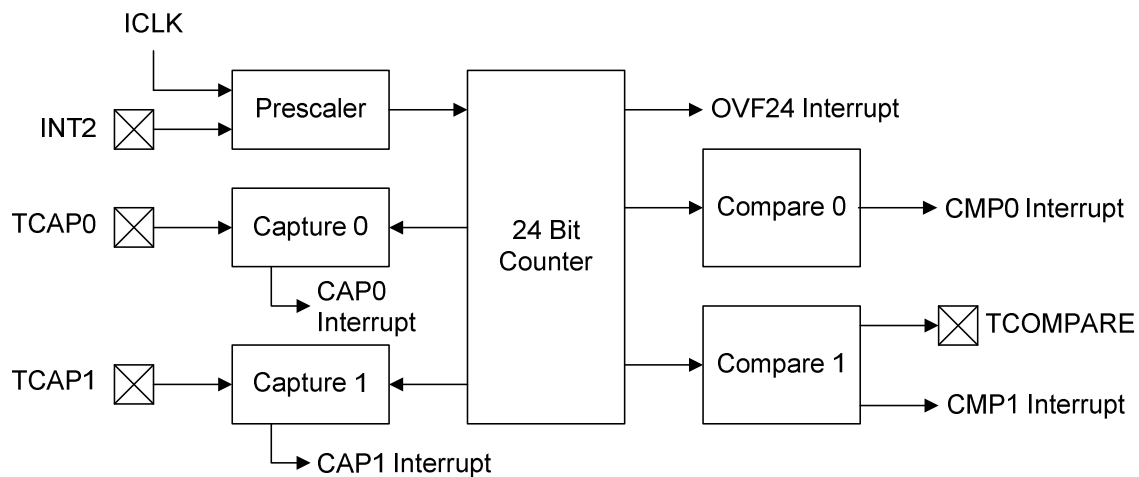
T24, a 24 bit free-running timer that is best used for timing asynchronous events

4 instances of T16PWM, a 16 bit timer best used for PWM and for generating regular, recurring interrupts.

WD – a watchdog timer.

1 T24 – 24 Bit Free-Running Timer with Capture and Compare

Here is a block diagram of the T24 Module:



1.1 T24 Clock Source, Prescaler and Counter

The core of the T24 timer is a 24 bit free-running counter. The clock input for the T24 timer can come from a clock within the UCD30xx, ICLK, which is nominally approximately 16 MHz. on the UCD3040. See the datasheet for the precise specification.

It can also be used with an external clock from the INT2 pin.

The internal clock is the default. To select the external clock source instead, execute this C equation:

```
TimerRegs.T24CNTCTRL.bit.EXT_CLK_SEL = 1;
```

All C code references in this document use standard TI header files. The first term, TimerRegs, refers to the Timer Register block.

The second term, T24CNTCTRL, refers to the specific register within that block.

Bit refers to the bit map. EXT_CLK_SEL refers to the specific bit field within the bit map.

These same names are used in the reference section in this document:

The clock then runs through a prescaler. The prescaler is controlled by an 8 bit register. Register values from 0 to 255 correspond to dividing the clock by 4 to 256.

The divider count always equals register +1, except for values below 3, which always result in a divider count of 4. The register C code is:

```
TimerRegs.T24CNTCTRL.bit.PRESCALE = 255;
```

With the nominal ICLK, and the 24 bit counter, a divide of 256 means that the timer will free run for about 4.5 minutes before overflowing.

Counter overflow can be used to generate an interrupt, if desired. The default is for the interrupt to be disabled. To enable it, here is the C code:

```
TimerRegs.T24CNTCTRL.bit.OV_INT_ENA = 1;
```

There is also a bit that can be polled to indicate that an overflow has occurred:

```
result = TimerRegs.T24CNTCTRL.bit.OV_FLAG;
```

It is necessary to write a 1 to the OV_FLAG to clear it. This will also clear the overflow interrupt.

The 24 bit free running counter can be read by firmware to provide a time base. For example, to measure the time required for a section of firmware, simply read the counter at the beginning and end of the section and subtract the start count from the end count. To read from the counter:

```
result = TimerRegs.T24CNTDAT.bit.CNT_DAT;
```

It is necessary to design the program to compensate for a counter overflow in case one occurs during that section of firmware. The OV_FLAG bit is useful in this case.

The counter is completely free running. There is no way to write to it, reset it, or to change when it overflows. It just keeps counting until all 24 bits overflow, then it starts over from zero.

1.2 T24 Capture Blocks

The T24 has two capture blocks, 0 and 1. The capture blocks are used to collect timing information about external signals. They can be programmed to capture the 24 bit timer value on an edge on the external signal. They can also generate an interrupt on the edge.

The edge control bits can select:

00 = No Capture (Default)
01 = Rising Edge
10 = Falling Edge
11 = Both Edges

For example, to program for both edges:

```
TimerRegs.T24CAP0CTRL.bit.EDGE = 3; //both edges
```

There are CAP_INT_ENA and CAP_INT_FLAG bits which work much the same as the overflow bits for the 24 bit counter. They are also in the T24CAP0CTRL register. For the Capture 1 block, there is a corresponding T24CAP1CTRL register.

For low speed signals, a single capture block can be used for both edges. For example, to determine the positive pulse width of a signal, connect the signal to the TCAP0 pin.

Enable capture on the rising edge

Poll the CAP_INT_FLAG bit, or enable the interrupt.

When the capture occurs, read the capture data register

```
result = TimerRegs.T24CAP0DAT.bit.CAP_DAT;
```

Then enable capture on the falling edge.

When the capture occurs, read the capture data register again and subtract:

```
result = TimerRegs.T24CAP0DAT.bit.CAP_DAT - result;
```

Note that overflow handling may be necessary as well. This depends on the expected timing of the signal.

One simple solution is that if the final result is negative, adding $0x1000000$, (2^{24}) will give the correct result. This is accurate so long as only one overflow has occurred. If there is a possibility of more than one overflow occurring, it will be necessary to keep track of this.

The same approach can apply to any combination of edges.

For measuring fast pulses, it may be necessary to use both capture pins for a single signal. For a fast positive pulse, for example, program one capture register for the rising edge and one for the falling edge.

Note that this only works when the low signal time is long enough to permit initialization. If the low going pulse is fast also, it is not possible to tell which edge came first.

There is also a T24CAPIO register which permits using the TCAP0 and TCAP1 pins as General Purpose I/O pins. It is very straightforward to use. See the Reference section at the end of this document for more information.

1.3 T24 Compare Blocks

The counter is also used by 2 compare blocks. Compare blocks are programmed with a 24 bit value. When the 24 bit counter matches that value, the compare block is triggered.

Compare 0 can only be used to trigger an interrupt. Compare 1 can also trigger an interrupt. It can also be programmed to set, reset, or toggle an external pin, the T_COMPARE pin.

These compare blocks are best used when the firmware starts an asynchronous process that needs some other event to follow at a fixed interval. Add the desired interval to the current counter value, and put the result into the compare register. When the time elapses, the interrupt will occur, or the TCMP pin will change.

To load the compare register with a fixed interval:

```
#define FIXED_INTERVAL 100
.....
```

```
TimerRegs.T24CMP0DAT.bit.CMP_DAT = TimerRegs.T24CNTDAT.bit.CNT_DAT + FIXED_INTERVAL;
```

This technique will even work well if an overflow occurs during the interval. Just like the timer, the result will overflow and be clipped.

The TCMP pin can also be used as a General Purpose I/O pin.

The configuration of the Compare Blocks is relatively straightforward. Refer to the Reference section below for details. One unusual feature is the use of the T_COMPARE pin as an output.

When the T_COMPARE pin is configured as an output, there are two sets of registers which control it. Whenever the compare 1 data register matches the 24 bit counter, the value of the pin is controlled by the OUT_ACTION bit field. This can be programmed for:

- 00 = Disabled (Default)
- 01 = Set pin
- 10 = Clear pin
- 11 = Toggle pin

The pin is also controlled by the OUT bit and the OUT_DRV bit. Whenever a 1 is written to the OUT_DRV bit, the pin is set to the value of the OUT bit. The OUT_DRV bit is self-clearing, so there is no need to write a zero to it before writing the next one.

This arrangement makes it possible to use both software and the compare logic to control the T_COMPARE bit in a seamless way.

1.4 T24 Interrupts

There are 5 T24 interrupts. Each one has a separate bit in the CIM – Central Interrupt Module. This means that each interrupt can have an independent interrupt vector, with no need to read from the timer module to determine which interrupt has occurred.

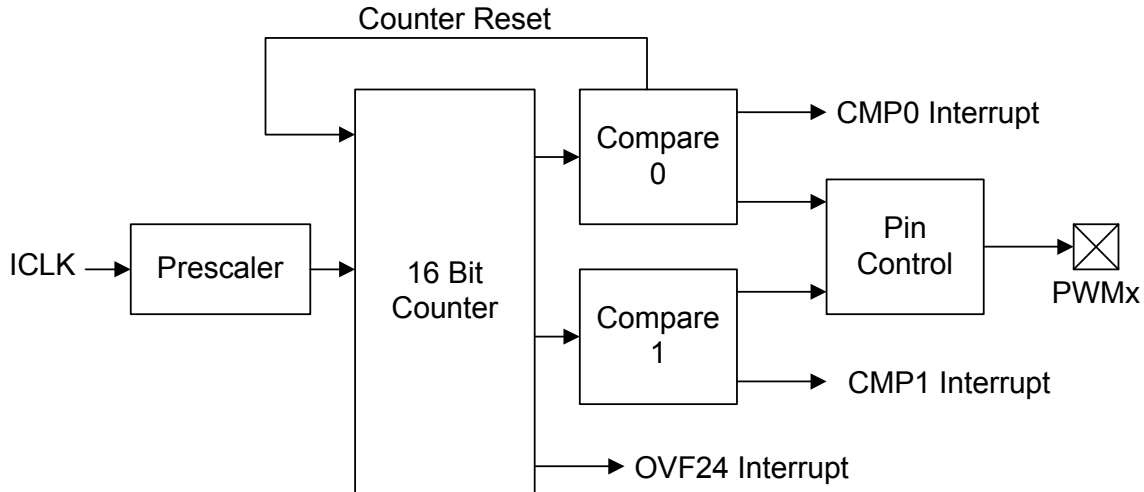
2 T16PWMx - 16 Bit PWM Timers

The UCD3040 has 4 independent 16 bit timers. These are best used for repetitive, regular interrupts, and for PWM generation, for example for fan drive control.

Other devices may have fewer timers. Consult their data sheets for specific information.

2.1 T16PWMx Summary

The 16PWMx timers have a 16 bit counter driven by a prescaler, which is driven by ICLK. They have 2 compare units which combine to control one output pin. Each compare unit can generate an interrupt, as can the counter overflow. All of the interrupts are combined into one before being sent to the CIM interrupt module.



2.2 T16PWMx Prescaler and Counter

The Prescaler for the 16 bit PWM Timer is always driven by ICLK. The clock divide is equal to PRESCALE + 1.

Here is C code which sets the prescale value to 3, for a divide of 4:

```
TimerRegs.T16PWM0CNTCTRL.bit.PRESCALE = 3; //divide by 4
```

To see the counter value, read this way:

```
result = TimerRegs.T16PWM0CNTDAT.bit.CNT_DAT;
```

If the timer overflows, it can generate an interrupt. The interrupt and flag bits are well described in the reference section below.

The timer can also be reset by the a compare event on the compare 0 block. To enable this function, use this C code:

```
TimerRegs.T16PWM0CNTCTRL.bit.CMP_RESET_ENA = 1; //enable Comp 0 reset
```

The 16 bit counter can also be reset by setting the SW_RESET bit:

```
TimerRegs.T16PWM0CNTCTRL.bit.SW_RESET = 0; //reset and stop counter
```

A zero in SW_RESET is the default state, so it is necessary to write a 1 to this bit in order for the counter to run.

2.3 T16PWMx Compare Blocks

There are two Compare Blocks for the T16PWMx timer. The Compare 0 block has the capability to reset the 16 bit counter. The Compare 1 block does not.

The Compare 0 block also has the highest priority. If both compare data registers are loaded with the same value, the Compare 0 operation will take place, and the Compare 1 operation will be ignored.

Each compare block has a 16 bit data register. When the 16 bit counter value matches the value in the data register, a programmed action can take place on the output pin for the T16. Here is the C code to write to a compare data register:

```
TimerRegs.T16PWM0CMP0DAT.bit.CMP_DAT = 100;
```

The compare action is programmable for each compare block to one of 4 states:

- 00 = No action (Default)
- 01 = Set pin
- 10 = Clear pin
- 11 = Toggle pin

To set Compare 0 up to toggle the pin:

```
TimerRegs.T16PWM0CMPCTRL.bit.OUT_ACTION0 = 3;
```

There is also an OUT bit and an OUT_DRV bit. Just like the 24 bit counter, when a 1 is written to the OUT_DRV bit, the OUT bit value is placed on the pin. Both compare blocks and the OUT bit can work seamlessly together.

All of this only works, of course, if the OUT_ENA bit is set as well. The default state for the pin is as an input, so it is necessary to set the OUT_ENA bit if output is desired.

The datasheet lists the pins as PWM1 through PWM4. If that is the case, pin PWM1 corresponds with T16PWM0, PWM2 with T16PWM1, and so on.

As already mentioned, each compare block can cause an interrupt, if it is enabled. The interrupt enable and flag bits are described in the reference section.

2.4 T16 Shadow Bit.

The compare control register also contains a SHADOW bit. It is normally cleared, but it can be set this way:

```
TimerRegs.T16PWM0CMPCTRL.bit.SHADOW = 1;
```

The SHADOW bit is useful for a PWM output that will vary over time. The compare blocks work by detecting when the compare data is equal to the 16 bit counter value. This works fine if the compare data is never changed.

If, however, the compare data value is reduced at the wrong time, it is possible for the compare block to miss it, and therefore for the PWM pulse to remain on for the entire period.

For example, suppose that the compare 1 data value is 100. Suppose that the counter value is 90, and the compare 1 data register is written to, changing the data value to 80. In this case, for one cycle, the compare 1 event will not occur. Normally for PWM, compare 1 is used to turn off the PWM pulse. So the result will be a PWM pulse that is on for the entire period.

If the SHADOW bit is set, it enables buffer registers, called shadow registers, for the compare data registers. The data written is stored in the shadow register until the next compare event, and only then is it written into the compare data register.

So to return to the example, the 80 will not be written into the data register until after the counter reaches 100, so the pulse will stop at 100, instead of going on to the end.

There are two side effects to using the shadow register:

1. Read always reads from the actual data register. So if the data register is written to and then read from before the next compare event for that block, the old data will still be read.
2. If the value written to the data register is higher than the current value, it may be possible to have two compare events. For example if the counter value is 70, the data value is 80, and the new data value is 100, the compare event will occur at 80, and the shadow register will be copied into the data register. Then a second compare event will occur when the 16 bit counter reaches 100. This should not be a problem if the compare action is a set or reset, but if the action is a toggle, this could be a problem. Or if the compare interrupt is being used.

Of course, the same thing could happen without the shadow bit set.

2.5 T16 Interrupts

The T16 module generates 3 interrupts. All of them are combined into a single signal for the CIM. Therefore, if more than one interrupt is enabled for a module, it will be necessary to read the module registers in order to determine the source(s) of the interrupt. It will also be necessary to write to all the interrupt flags which are set to clear the interrupt.

2.6 Using the T16 for a Timer Interrupt

The C code below should provide an approximate 10 KHz interrupt from the T16:

```
TimerRegs.T16PWM0CMP0DAT.bit.CMP_DAT = 1587; //value to reset counter
TimerRegs.T16PWM0CMPCTRL.bit.CMP0_INT_ENA = 1; //interrupt when counter reset
TimerRegs.T16PWM0CNTCTRL.bit.CMP_RESET_ENA = 1; //enable reset by comp 0
TimerRegs.T16PWM0CNTCTRL.bit.SW_RESET = 1; //allow counter to run
```

2.7 Using the T16 for PWM generation

The C code below should provide a 50% PWM at approximately 10 KHz. Consult the datasheet for ICLK speeds.

```
TimerRegs.T16PWM0CMP0DAT.bit.CMP_DAT = 1587; //value to reset counter
TimerRegs.T16PWM0CMP1DAT.bit.CMP_DAT = 793; //50% duty cycle half of comp 0
TimerRegs.T16PWM0CMPCTRL.bit.OUT_ACTION0 = 1; //1 is for clear pin
TimerRegs.T16PWM0CMPCTRL.bit.OUT_ACTION1 = 2; //2 is for clear pin
TimerRegs.T16PWM0CNTCTRL.bit.CMP_RESET_ENA = 1; //enable reset by comp 0
TimerRegs.T16PWM0CMPCTRL.bit.OUT = 0; //make sure that default is a 0
TimerRegs.T16PWM0CMPCTRL.bit.OUT_DRV = 1; //put zero into output latch
TimerRegs.T16PWM0CMPCTRL.bit.OUT_ENA = 1; //enable pin as an output
TimerRegs.T16PWM0CNTCTRL.bit.SW_RESET = 1; //allow counter to run
```

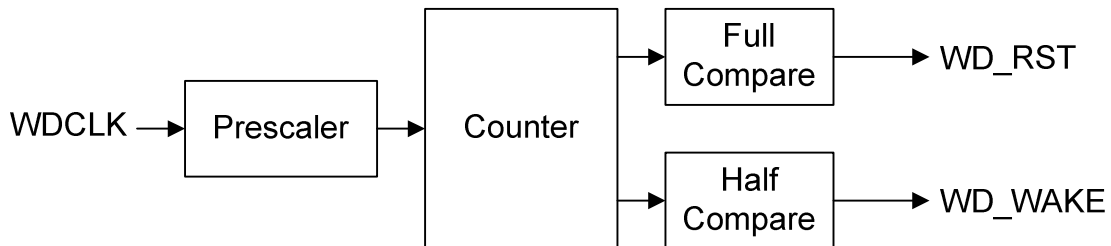
This program uses the Capture 0 block to set the PWM pin high at the end of the previous cycle and simultaneously reset the counter. Then the Capture 1 block, which is loaded with a value 50% the size of Capture 0, clears the PWM pin value halfway through the period.

3 WD – Watchdog

The watchdog is the smallest module in the timer. It is primarily designed to assist in recovery in the event of a firmware or processor fault. The watchdog is not intended for use as a source of a timer interrupt.

Unlike the other timers, the counter in the watchdog is invisible, and the compare data values are fixed.

The interrupts occur always when the counter is half full and when it is full.



3.1 Watchdog prescale and counter.

The watchdog "prescale" is the PERIOD bit field. It sets the counter period for the watchdog to a time between approximately 10 milliseconds and approximately 1.3 seconds.

The counter can be reset by writing a 1 to the CNT_RESET bit.

The counter output is used by two compare blocks with fixed values.

The Watchdog Timer is driven by its own unique oscillator. It will be triggered even if the main clock oscillator fails.

3.2 Watchdog compare blocks

The two compare blocks are one for the counter half full, and one for the counter overflow.

The half full counter interrupt is intended to give advance warning that something is wrong with program execution. If the interrupt function is still working, it can be used to start a recovery operation.

The counter overflow can also be used just as an interrupt, or it can be configured to reset the CPU.

The configuration is relatively simple, and can be understood from the Reference section below.

Note that the Watchdog Wake Event is another name for the Half Compare Event.

The only special bit is the Protect bit.

3.3 Watchdog Protect Bit

The Watchdog Protect bit is a special bit. Once it is cleared, it cannot be set again. It also causes other bits to be set so that they cannot be cleared. The effect is that once the protect bit is cleared, the watchdog timer cannot be turned off. It will always need to be cleared by writing a 1 to CNT_RESET.

Other bits in the Watchdog Control register are not protected, however. For example, the WD_PERIOD bits can still be changed, as can the interrupt enable bits. See the reference section below for details

3.4 Watchdog Timer Example

In normal watchdog operation, the watchdog is set up to reset the CPU if it makes it to a full count. The firmware is designed to write a 1 to the CNT_RESET bit frequently, so that the watchdog should never reach a full count unless some fault causes the firmware to stop executing properly.

Initialization of the watchdog is very simple:

```
TimerRegs.WDCTRL.bit.CNT_RESET = 1; //make sure counter is cleared
TimerRegs.WDCTRL.bit.PROTECT = 0; //enable protected watchdog
```

Then just repeat the statement from above frequently enough in the code to prevent the watchdog counter from overflowing:

```
TimerRegs.WDCTRL.bit.CNT_RESET = 1; //make sure counter is cleared
```

If a faster watchdog timeout is desired, simply write to the WD_PERIOD bitfield:

```
TimerRegs.WDCTRL.bit.WD_PERIOD = 2; //fast watchdog. Greyhound?
```

The default value for WD_PERIOD is 0x7f, which will give an approximate 1.3 second watchdog.

The initial CNT_RESET above is necessary because the Watchdog counter is free-running. It is not stopped by any bits, so it could be very close to overflow when the Watchdog is initialized. The write to CNT_RESET ensures that it is started at 0 upon initialization.

3.5 Warnings for Watchdog Status Register

The Watchdog Status Register is a clear on read register. The Watchdog is asynchronous to the CPU. On rare instances, it is possible to read from the register at exactly the right time and read a 0 even though the bit has been set and cleared.

To avoid this issue, set the interrupt enable bits in the WDCTRL register, even if the interrupts are not going to be used. Then poll the interrupt bits in the CIM to determine when the WD bits are set. Then it is safe to read from the WD status register to clear the bits, which will also clear the CIM bits.

Note also that the WD_HALF_FLAG will be set at the halfway point and will be set again at around the same time as the WDRST bit is set.

Also note that all three bits in the WDST register may be set upon power up reset, so they should be read from in order to clear them before any polling operation begins.

Also note, of course, that any bitwise read actually reads from the entire register and will clear all the bits which are set. So if sampling of multiple bits is desired, copy the register into a variable and examine the bits individually there.

3.6 *Metastable states in the Watchdog Logic*

Occasional metastable state issues have been observed in the Watchdog logic. These may cause the watchdog to fail to time out.

Normally, the CNT_RESET bit is automatically cleared by the Watchdog logic. If the CNT_RESET bit develops a metastable state, it can fail to be cleared by the logic, so the Watchdog counter will continue to be reset.

To avoid this issue, after every

TimerRegs.WDCTRL.bit.CNT_RESET = 1; //make sure counter is cleared

statement, issue the following statement to clear the CNT_RESET:

TimerRegs.WDCTRL.bit.CNT_RESET = 0; //make sure counter clear bit is cleared

This statement cannot immediately follow the first one. The Watchdog timer has its own clock, which may run as low as 20 KHz. It is necessary to allow 2 clock cycles for the CNT_RESET to take effect.

Therefore, there must be at least a 100 microsecond delay between the first statement and the second one. Otherwise the watchdog will not be reset reliably. The CNT_RESET bit could be reset before it can reset the watchdog counter.

This delay can be as simple as a loop of nop instructions, or it can be done in an interrupt.

3.7 *System Fault Recovery Basics*

There is a register in the System Module – SYSESR – System Exception Status Register – which saves the reset cause through the reset. There is a bit for every event except the watchdog timer.

It is possible to look at this register when the program is entered and determine if the reset was caused by the watchdog timer, or some other cause, such as an illegal address access. Often if a problem occurs with firmware execution, some other fault will occur before the watchdog timer times out. So if a reset occurs and none of the bits are set, then the cause is the watchdog timer.

To use this register, examine it when the program is first started. Do whatever processing is necessary, save the reset cause if desired, and then clear the register. In this way, it will be possible to tell the next reset cause. If the register is not cleared by the program, there may be two bits set the next time, making it impossible to determine the cause of the newest reset.

All resets, regardless of the cause, will reset all of the peripherals to their default state, so most outputs will go to inputs and so on. This will generally have the effect of shutting down any power supply that is being controlled by the device.

4 Timer Module Register Reference

All addresses in this section are correct for the UCD3040. Other devices may have different addresses and some peripherals may not be available. Please consult the data sheet for the specific device for more information.

Timer Registers have the following attributes:

- 32-bit wide
- Addresses placed on word boundaries
- Byte, Half-Word and word writes permitted
- All Registers can be read in any mode
- All Registers, except for the Timer Powerdown Control Register, are writeable in any mode. The Timer Powerdown Control Register is writeable only in privilege mode.

4.1 24-bit Counter Data Register (T24CNTDAT)

Address FFF7FD00

Bit Number	23:0
Bit Name	CNT_DAT
Access	R
Default	-

Bits 23-0: CNT_DAT: – Contains the 24-bit counter value

4.2 24-bit Counter Control Register (T24CNTCTRL)

Address FFF7FD04

Bit Number	15:8	7:3	2	1	0
Bit Name	PRESCALE	RESERVED	EXT_CLK_SEL	OV_INT_ENA	OV_FLAG
Access	R/W	-	R/W	R/W	R
Default	0000_0000	00000	0	0	0

Bits 15-8: PRESCALE – Defines the prescaler value used to select the 24-bit counter resolution. The minimum divider ratio is 4, prescaler value less than 3 defaults to 3.

Counter Resolution = (Prescaler Value+1)*1/ICLK

Bits 7-3: RESERVED – Unused bits

Bit 2: EXT_CLK_SEL – External Clock Select

0 = Selects ICLK as clock for 24-bit counter (Default)

1 = Selects External Clock on INT_2 as clock for 24-bit counter

Bit 1: OV_INT_ENA – Counter Overflow Interrupt Enable

0 = Disables 24-bit Counter Overflow Interrupt (Default)

1 = Enables 24-bit Counter Overflow Interrupt

Bit 0: OV_FLAG – Indicates a counter overflow. Overflow event is cleared by writing a '1' to this bit. If a clear and an overflow event occur at the same time, the flag will remain high (set has priority versus clear).

0 = No counter overflow since last clear

1 = Counter overflow since last clear

4.3 24-bit Capture Channel 0-1 Data Register (T24CAPxDAT)

Address FFF7FD08 – 24-bit Capture Channel 0 Data Register

Address FFF7FD0C – 24-bit Capture Channel 1 Data Register

Bit Number	23:0
Bit Name	CAP_DAT
Access	R
Default	-

Bits 23-0: CAP_DAT– Contains the 24-bit input capture value

4.4 24-bit Capture Channel 0-1 Control Register (T24CAPxCTRL)

Address FFF7FD14 – 24-bit Capture Channel 0 Control Register

Address FFF7FD18 – 24-bit Capture Channel 1 Control Register

Bit Number	3:2	1	0
Bit Name	EDGE	CAP_INT_ENA	CAP_INT_FLAG
Access	R/W	R/W	R/W
Default	00	0	0

Bits 3-2: EDGE – Input Capture Edge Select

00 = No Capture (Default)

01 = Rising Edge

10 = Falling Edge

11 = Both Edges

Bit 1: CAP_INT_ENA – Input Capture Interrupt Enable

0 = Disables 24-bit input capture interrupt (Default)

1 = Enables 24-bit input capture interrupt

Bit 0: CAP_INT_FLAG – Flag which indicates a valid input capture event. This bit is cleared by writing a '1' to it or by reading the corresponding Capture Channel Data Register. If a clear and a valid capture event occur at the same time, the flag will remain high (set has priority versus clear).

0 = No valid capture event since last clear

1 = Valid capture event since last clear

4.5 24-bit Capture I/O Control and Data Register (T24CAPIO)

Address FFF7FD20

Bit Number	5	4
Bit Name	DIN1	DOUT1
Access	R	R/W
Default	-	0

Bit Number	3	2	1	0
Bit Name	DIR1	DIN0	DOUT0	DIR0
Access	R/W	R	R/W	R/W
Default	0	-	0	0

Bit 5: DIN1 – Input data for pin TMR_TCAP1

0 = Logic level low detected on TMR_TCAP1

1 = Logic level high detected on TMR_TCAP1

Bit 4: DOUT1 – Output data for pin TMR_TCAP1

0 = Logic level low driven on TMR_TCAP1 when configured as output (Default)

1 = Logic level high driven on TMR_TCAP1 when configured as output

Bit 3: DIR1 – Controls data direction for pin TMR_TCAP1

0 = TMR_TCAP1 pin configured as input (Default)

1 = TMR_TCAP1 pin configured as output

Bit 2: DIN0 – Input data for pin TMR_TCAP0

0 = Logic level low detected on TMR_TCAP0

1 = Logic level high detected on TMR_TCAP0

Bit 1: DOUT0 – Output data for pin TMR_TCAP0

0 = Logic level low driven on TMR_TCAP0 when configured as output (Default)

1 = Logic level high driven on TMR_TCAP0 when configured as output

Bit 0: DIR0 – Controls data direction for pin TMR_TCAP0

0 = TMR_TCAP0 pin configured as input (Default)

1 = TMR_TCAP0 pin configured as output

4.6 24-bit Output Compare Channel 0-1 Data Register (T24CMPxDAT)

Address FFF7FD24 – 24-bit Output Compare Channel 0 Data Register

Address FFF7FD28 – 24-bit Output Compare Channel 1 Data Register

Bit Number	23:0
Bit Name	CMP_DAT
Access	R/W
Default	0000_0000_0000_0000_0000_0000

Bits 23-0: **CMP_DAT**– Contains the 24-bit output comparison value

4.7 24-bit Output Compare Channel 0 Control Register (T24CMP0CTRL)

Address FFF7FD2C

Bit Number	1	0
Bit Name	CMP0_INT_ENA	CMP0_INT_FLAG
Access	R/W	R/W
Default	0	0

Bit 1: CMP0_INT_ENA– Output Compare Channel 0 Interrupt

0 = Disables Output Compare Channel 0 Interrupt (Default)

1 = Enables Output Compare Channel 0 Interrupt

Bit 0: CMP0_INT_FLAG – Indicates a valid output compare event. Bit can be cleared by writing a '1' to the bit or by rewriting the 24-bit Output Compare Channel 0 Data Register. If a clear and compare event occur at the same time, the flag will remain high (set has priority versus clear).

0 = No compare event since last clear

1 = Compare event since last clear

4.8 24-bit Output Compare Channel 1 Control Register (T24CMP1CTRL)

Address FFF7FD30

Bit Number	7	6	5	4
Bit Name	IN	OUT	OUT_ENA	OUT_DRV
Access	R	R/W	R/W	R/W
Default	-	0	0	0

Bit Number	3:2	1	0
Bit Name	OUT_ACTION	CMP1_INT_ENA	CMP1_INT_FLAG
Access	R/W	R/W	R/W
Default	0	0	0

Bit 7: IN – Input Value of TMR_TCMP pin

0 = Logic level low detected on TMR_TCMP pin

1 = Logic level high detected on TMR_TCMP pin

Bit 6: OUT – Contains the data to be written into the output latch when ODRV is high.

0 = Output latch is cleared when ODRV = 1

1 = Output latch is set when ODRV = 1

Bit 5: OUT_ENA – TMR_TCMP Pin Configuration

0 = TMR_TCMP configured as an input pin (Default)

1 = TMR_TCMP configured as an output pin

Bit 4: OUT_DRV – Causes the value of the bit OD to be written into the output latch. It is possible to preload the output latch or to use the pin as GPIO. The compare action has priority before the preload function. The bit is always read as '0'.

0 = Output latch not affected by the value of OD (Default)

1 = Value of OD written into the output latch

Bit 3-2: OUT_ACTION – These 2 bits select the output action when a compare equal state is detected.

00 = Disabled (Default)

01 = Set pin

10 = Clear pin

11 = Toggle pin

Bit 1: CMP1_INT_ENA – Output Compare Channel 1 Interrupt Enable

0 = Disables Output Compare Channel 1 Interrupt (Default)

1 = Enables Output Compare Channel 1 Interrupt

Bit 0: CMP1_INT_FLAG – Indicates a valid output compare event. Bit can be cleared by writing a '1' to the bit or by rewriting the 24-bit Output Compare Channel 1 Data Register. If a clear and compare event occur at the same time, the flag will remain high (set has priority versus clear).

0 = No compare event since last clear

1 = Compare event since last clear

4.9 PWMx Counter Data Register (T16PWMxCNTDAT)

Address FFF7FD34 – 16-bit PWM0 Counter Data Register

Address FFF7FD58 – 16-bit PWM1 Counter Data Register

Address FFF7FD6C – 16-bit PWM2 Counter Data Register

Address FFF7FD80 – 16-bit PWM3 Counter Data Register

Bit Number	15:0
Bit Name	CNT_DAT
Access	R
Default	-

Bits 15-0: CNT_DAT – Contains the 16-bit counter value. Read-only.

4.10 PWMx Counter Control Register (T16PWMxCNTCTRL)

Address FFF7FD38 – 16-bit PWM0 Counter Control Register

Address FFF7FD5C – 16-bit PWM1 Counter Control Register

Address FFF7FD70 – 16-bit PWM2 Counter Control Register

Address FFF7FD84 – 16-bit PWM3 Counter Control Register

Bit Number	15:8	7:4	3
Bit Name	PRESCALE	RESERVED	SW_RESET
Access	R/W	-	R/W
Default	0000_0000	-	0

Bit Number	2	1	0
Bit Name	CMP_RESET_ENA	OV_INT_ENA	OV_INT_FLAG
Access	R/W	R/W	R/W
Default	0	0	0

Bit 15-8: PRESCALE – Defines the prescaler value to select the PWM counter resolution.

Counter Resolution = (Prescaler + 1) * 1/ICLK

Bit 3: SW_RESET– PWM counter reset by software. This bit is cleared after reset and has to be set to run the PWM counter.

0 = PWM counter reset and counter stop (Default)

1 = PWM counter is running

Bit 2: CMP_RESET_ENA – Enables PWM counter reset by compare action of Compare 0.

0 = Disable PWM counter reset by Compare 0 action (Default)

1 = Enable PWM counter reset by Compare 0 action

Bit 1: OV_INT_ENA– PWM Counter Overflow Interrupt Enable

0 = Disable PWM counter overflow interrupt (Default)

1 = Enable PWM counter overflow interrupt

Bit 0: OV_INT_FLAG – Flag which indicates a PWM counter overflow. This bit is cleared by writing '1' to it. If a clear and an overflow event occur at the same time, the flag will remain high (set has priority versus clear).

0 = No PWM counter overflow since last clear

1 = PWM counter overflow since last clear

4.11 PWMx 16-bit Compare Channel 0-1 Data Register (T16PWMxCMPyDAT)

Address FFF7FD3C – 16-bit PWM0 Compare Channel 0 Data Register

Address FFF7FD40 – 16-bit PWM0 Compare Channel 1 Data Register

Address FFF7FD60 – 16-bit PWM1 Compare Channel 0 Data Register

Address FFF7FD64 – 16-bit PWM1 Compare Channel 1 Data Register

Address FFF7FD74 – 16-bit PWM2 Compare Channel 0 Data Register

Address FFF7FD78 – 16-bit PWM2 Compare Channel 1 Data Register

Address FFF7FD88 – 16-bit PWM3 Compare Channel 0 Data Register

Address FFF7FD8C – 16-bit PWM3 Compare Channel 1 Data Register

Bit Number	15:0
Bit Name	CMP_DAT
Access	R/W
Default	0000_0000_0000_0000

Bits 15-0: CMP_DAT – Contains the 16-bit compare value. When in PWM mode, the value in the T16PWMxCMPyDAT is loaded after a match with the PWMx Counter Data Register. When in OC mode, it has to be written by the CPU. The mode is controlled by the bit SHADOW in the PWMx/Dual Compare Control Register. If both Registers T16PWMxCMP0DAT and T16PWMxCMP1DAT contain the same value, the interrupt and pin behavior is controlled by output compare channel 0 (T16PWMxCMP0DAT has priority over T16PWMxCMP1DAT).

4.12 PWMx Compare Control Register (T16PWMxCMPCTRL)

Address FFF7FD44 – 16-bit PWM0 Compare Control Register

Address FFF7FD68 – 16-bit PWM1 Compare Control Register

Address FFF7FD7C – 16-bit PWM2 Compare Control Register

Address FFF7FD90 – 16-bit PWM3 Compare Control Register

Bit Number	12	11	10	9	8
Bit Name	SHADOW	IN	OUT	OUT_ENA	OUT_DRV
Access	R/W	R	R/W	R/W	R/W
Default	0	-	0	0	0

Bit Number	7:6	5:4	3	2
Bit Name	OUT_ACTION1	OUT_ACTION0	CMP1_INT_ENA	CMP1_INT_FLAG
Access	R/W	R/W	R/W	R/W
Default	00	00	0	0

Bit Number	1	0
Bit Name	CMP0_INT_ENA	CMP0_INT_FLAG
Access	R/W	R/W
Default	0	0

Bit 12: SHADOW – Controls the update of the 16-bit output compare Registers.

0 = PWM output compare Registers immediately written (Default)

1 = PWM output compare Registers updated through the buffers T16PWMxCMPyDAT after a match occurs in the corresponding Register T16PWMxCMPyDAT.

Bit 11: IN – Input value of TMR_TPWM pin

0 = Logic level low detected on TMR_TPWM pin

1 = Logic level high detected on TMR_TPWM pin

Bit 10: OUT – Data to be written into the output latch when OUT_DRV is high.

0 = Output latch is cleared when OUT_DRV=1 (Default)

1 = Output latch is set when OUT_DRV=1

Bit 9: OUT_ENA – TMR_TPWM pin configuration

0 = TMR_TPWM configured as an input pin (Default)

1 = TMR_TPWM configured as an output pin

Bit 8: OUT_DRV – Causes the value of the bit OUT to be written into the output latch. So it is possible to preload the output latch or to use the pin as GPIO. The compare action has priority before the preload function. This bit is always read as '0'.

0 = Output latch not affected by the value of OUT (Default)

1 = Value of OUT written into the output latch

Bits 7-6: OUT_ACTION1 – These 2 bits select the output action when a compare equal is detected on T16CMP1DAT

00 = No action (Default)

01 = Set pin

10 = Clear pin

11 = Toggle pin

Bits 5-4: OUT_ACTION0 – Selects the output action when a compare equal is detected on T16CMP0DAT.

00 = No action (Default)

01 = Set pin

10 = Clear pin

11 = Toggle pin

Bit 3: CMP1_INT_ENA – Compare 1 Interrupt Enable

0 = Disables Compare 1 Interrupt (Default)

1 = Enables Compare 1 Interrupt

Bit 2: CMP1_INT_FLAG – Flag which indicates a valid output compare 1 event. This bit is cleared by

writing '1' to this bit or by rewriting T16PWMxCMP1DAT. If a clear and a compare event occurs at the same time, the flag will remain high (set has priority versus write clear).

0 = No compare event since last clear

1 = Compare event since last clear

Bit 1: CMP0_INT_ENA – Compare 0 Interrupt Enable

0 = Disables Compare 0 Interrupt (Default)

1 = Enables Compare 0 Interrupt

Bit 0: CMP0_INT_FLAG – Flag which indicates a valid output compare 1 event. This bit is cleared by writing '1' to this bit or by rewriting T16PWMxCMP0DAT. If a clear and a compare event occurs at the same time, the flag will remain high (set has priority versus write clear).

0 = No compare event since last clear

1 = Compare event since last clear

4.13 Watchdog Status (WDST)

Address FFF7FD94

Bit Number	2	1	0
Bit Name	WD_FLAG	WDRST	WD_HALF_FLAG
Access	R	R	R
Default	-	-	-

Bit 2: WD_FLAG – Watchdog Timer Flag

0 = Watchdog Timer has not fired

1 = Watchdog Timer has fired

Bit 1: WDRST – Captured watchdog reset output

0 = Watchdog reset has not been enabled

1 = Watchdog reset has been enabled

Bit 0: WD_HALF_FLAG – ½ Watchdog Timer Flag

0 = Watchdog Timer has not reached ½ level

1 = Watchdog Timer has reached ½ level

4.14 Watchdog Control (WDCTRL)

Address FFF7FD98

Bit Number	14:8	7	6	5	4
Bit Name	PERIOD	RESERVED	PROTECT	CPU_RESET_EN	WDRST_INT_EN
Access	R/W	-	R/W	R/W	R/W
Default	111_1111	0	1	0	1

Bit Number	3	2	1	0
Bit Name	WKEV_INT_EN	WDRST_EN	WKEV_EN	CNT_RESET
Access	R/W	R/W	R/W	R/W
Default	1	1	1	0

Bits 14-8: PERIOD - Configures the time for the watchdog reset.

H'7F ~ 1.3s (minimum) (Default)

H'00 ~ 10ms (maximum)

Bit 7: RESERVED – Unused bits

Bit 6: PROTECT – Watchdog Protect Bit, Active Low

0 = Watchdog enable bits are protected, only can be cleared by POR. CPU_RESET_ENA (Bit 5), WDRST_ENA (Bit 2) and WKEV_ENA (Bit 1) are automatically set high when PROTECT is

written low.

1 = Watchdog enable bits can be set by processor (Default)

Bit 5: CPU_RESET_ENA – Enables Watchdog Reset Event to reset the CPU

0 = Watchdog Reset does not reset CPU (Default)

1 = Watchdog Reset does resets CPU

Bit 4: WDRST_INT_ENA – Watchdog Reset Event Interrupt Enable

0 = Disables generation of Watchdog Reset Interrupt

1 = Enables generation of Watchdog Reset Interrupt (Default)

Bit 3: WKEV_INT_ENA – Watchdog Wake Event Interrupt Enable

0 = Disables generation of Watchdog Wake Event Interrupt

1 = Enables generation of Watchdog Wake Event Interrupt (Default)

Bit 2: WDRST_ENA – Watchdog Reset Event Comparator Enable

0 = Disables Watchdog Reset Event Comparator

1 = Enables Watchdog Reset Event Comparator (Default)

Bit 1: WKEV_ENA – Watchdog Wake Event Comparator Enable

0 = Disables Watchdog Wake Event Comparator

1 = Enables Watchdog Wake Event Comparator (Default)

Bit 0: CNT_RESET – This bit resets the watchdog counters. This bit self clears and if the enables are set, the counters restart counting.

0 = Watchdog counters enabled (Default)

1 = Watchdog counters reset

5 Other Peripheral Registers Mentioned

5.1 System Exception Status Register (SYSESR)

Address FFFFFFFE4

The System Exception Status Register contains flags for different reset/abort sources. On power-up, all bits are cleared to 0. When a reset condition is recognized, the appropriate bit in the Register is set and the value of the bit is maintained through the reset. When a new reset condition occurs, the current contents of this Register are not cleared. The contents of this Register are cleared on a power-on reset or by software.

Bit Number	15	14	13	12	11	10
Bit Name	PORRST	CLKRST	RSVD	ILLMODE	ILLADR	ILLACC
Access	R/W	R/W	-	R/W	R/W	R/W
Default	0	0	0	0	0	0

Bit Number	9	8	7	6:0
Bit Name	PILLACC	ILLMAP	SWRST	RESERVED
Access	R/W	R/W	R/W	-
Default	0	0	0	000_0000

Bit 15: PORRST – Power-On reset flag. Set when power-on reset is asserted. Reset is asserted as long as power-on-reset is active. Whenever a device is powered, this bit is set.

User and privilege modes (read)

0 = Power-up reset has not occurred since the last clear

1 = Power-up reset has occurred since the last clear

User and privilege modes (write)

0 = Clears the corresponding bit to 0

1 = No effect

Bit 14: CLKRST – This bit represents the clock fail flag. This bit indicates a clock fault condition has occurred. After power-on-reset, the CLKRST is reset to 0. Value remains unchanged during other resets.

User and privilege modes (read)

0 = Clock failure has not occurred since the last clear

1 = Clock failure has occurred since the last clear

User and privilege modes (write)

0 = Clears the corresponding bit to 0

1 = No effect

Bit 13: RSVD – Reserved for future use

Bit 12: ILLMODE – This bit represents the illegal mode flag. This bit is set when the mode bits in the program status Register are set to an illegal value.

User and privilege modes (read)

0 = Illegal mode has not occurred since the last clear

1 = Illegal mode has occurred since the last clear

User and privilege modes (write)

0 = Clears the corresponding bit to 0

1 = No effect

Bit 11: ILLADR – This bit represents the illegal address access flag. This bit is set when an access to an unimplemented location in the memory map is detected in non-user mode.

User and privilege modes (read)

0 = Illegal address has not occurred since the last clear

1 = Illegal address has occurred since the last clear

User and privilege modes (write)

0 = Clears the corresponding bit to 0

1 = No effect

Bit 10: ILLACC – This bit represents the illegal memory access flag. This bit is set when an access to a protected location without permission rights is detected in non-user mode.

User and privilege modes (read)

0 = Illegal memory access has not occurred since the last clear

1 = Illegal memory access has occurred since the last clear

User and privilege modes (write)

0 = Clears the corresponding bit to 0

1 = No effect

Bit 9: PILLACC – This bit represents the peripheral illegal access flag. This bit is set when a peripheral access violation is detected in user mode.

User and privilege modes (read)

0 = Illegal peripheral access has not occurred since the last clear

1 = Illegal peripheral access has occurred since the last clear

User and privilege modes (write)

0 = Clears the corresponding bit to 0

1 = No effect

Bit 8: ILLMAP – This bit represents the illegal address map flag. This bit is set when the base addresses of one or more memories overlap. Reset occurs when the overlapped registration is accessed.

User and privilege modes (read)

0 = Illegal address mapping has not occurred since the last clear

1 = Illegal address mapping has occurred since the last clear

User and privilege modes (write)

0 = Clears the corresponding bit to 0

1 = No effect

Bit 7: SWRST – This bit represents the software reset flag. This bit is set when the last reset is caused by software writing the RESET bits.

User and privilege modes (read)

0 = Software reset has not occurred since the last clear

1 = Software reset has occurred since the last clear

User and privilege modes (write)

0 = Clears the corresponding bit to 0

1 = No effect

Bit 6-0: RESERVED