Application Report
SLAA601–October 2013

Using the MSP430 Timer_D Module in Hi-Resolution Mode

Damian Szmulewicz

ABSTRACT

This application report describes the use of the high-resolution feature of the Timer_D module introduced in MSP430F51x1 and MSP430F51x2 devices. Timer_D enables applications that require high resolution such as capacitive touch detection [1], PWM DACs [2] , advanced LED lighting, and digital controllers for power supplies [3]. This application report provides an advanced description of the use and calibration of Timer_D in high-resolution mode. For a basic description of Timer_D and its registers, see the *Timer_D* chapter from the *MSP430x5xx and MSP430x6xx Family User's Guide* (SLAU208). The examples shown in this document refer to the data sheet of the MSP430F5172 microcontroller [4], but the concepts are applicable to all devices with a Timer_D module. When working with this document, refer to the device-specific data sheet for all specifications.

Project collateral and source code discussed in this document are available for download from http://www.ti.com/lit/zip/slaa601.

Contents

List of Figures

List of Tables

1 Introduction

Timer_D is the third generation MSP430 timer module. Table 1 summarizes the main differences between Timer_D and its predecessors Timer_A and Timer_B. Timer_D includes a hi-resolution clock generator that can generate timer clock frequencies up to 256 MHz.

Table 1. Basic Comparison Among MSP430 General-Purpose Timers

Features	Timer_A	Timer_B	Timer_D
Max Timer Clock Frequency	25 MHz	25 MHz	256 MHz
Capture Mode	Single capture mode only	Single capture mode only	Single and dual capture mode
Dual Edge PWM Control (dead band generation)	X	X	Rising, falling or rising and falling edge control
Timer Control via External Events	X	X	Yes (Timer event control mode)
Synchronize Multiple Timer Instances	X	X	Yes (Timer event control mode)
Synchronized Loading of Compare Registers	X	Yes (group latching)	Yes (group latching)
LPMx Operation Without External Crystal	X	X	Yes (in free-running mode)
Enable Digital Power Factor Correction	X	X	Yes (shortening TEC events)

This application report focuses on the high-resolution features introduced to MSP430 with Timer_D. For the timer's functionality in normal mode and for a description of all registers, see *Timer_D Module (Chapter Excerpt From MSP430x5xx Family, SLAU208)* (SLAU232). In addition, this document is designed to complement, and not replace, the *MSP430x5xx and MSP430x6xx Family User's Guide* (SLAU208) and device-specific data sheets.

2 TIMER_D High Resolution Mode Overview

Timer_D is equipped with a high-resolution generation block, as shown in Figure 1, that achieves a maximum frequency of 256 MHz.

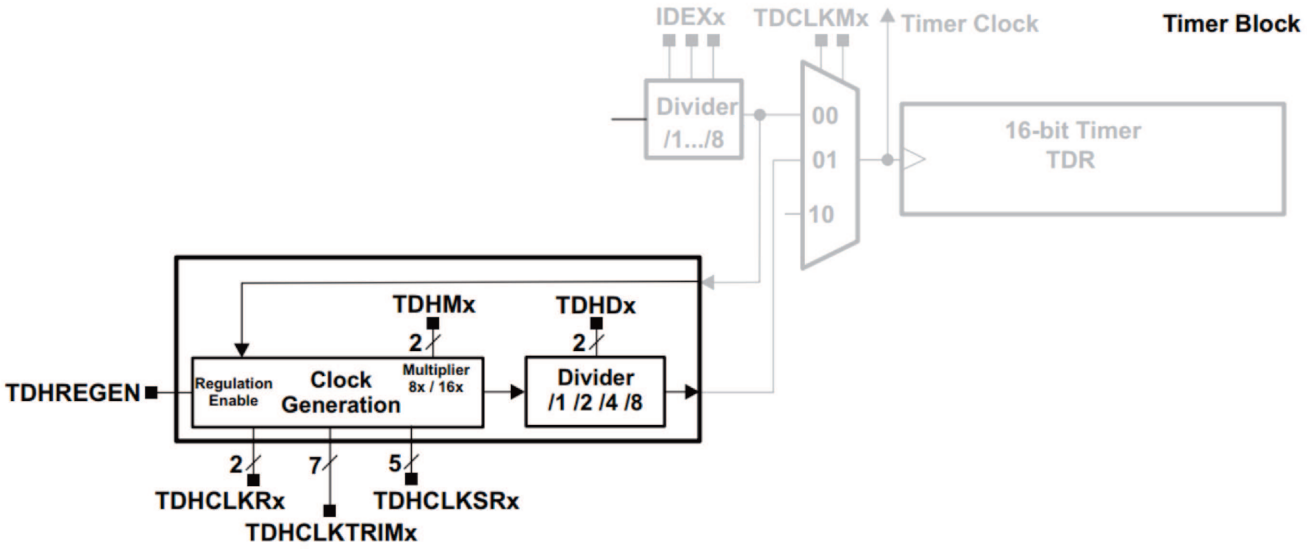

Figure 1. Timer_D High-Resolution Block

The high-resolution feature of Timer_D is enabled by setting the TDHEN bit in the TDxHCTL0 register. When TDHEN = 0, the timer runs in normal mode and its functionality is similar to Timer_A and Timer_B with some enhancements. For more information on the functionality of Timer_D for TDHEN = 0, see the *MSP430x5xx and MSP430x6xx Family User's Guide* (SLAU208). When TDHEN = 1, the timer is in high-resolution mode, and it operates in either Regulated or Free-running mode as shown in Figure 2. Regulation is enabled by setting TDHREGEN = 1 in TDxHCTL0 and it is disabled by clearing TDHREGEN = 0 in TDxHCTL0. The following sections describe the functionality of Timer_D in each of the presented operation modes.

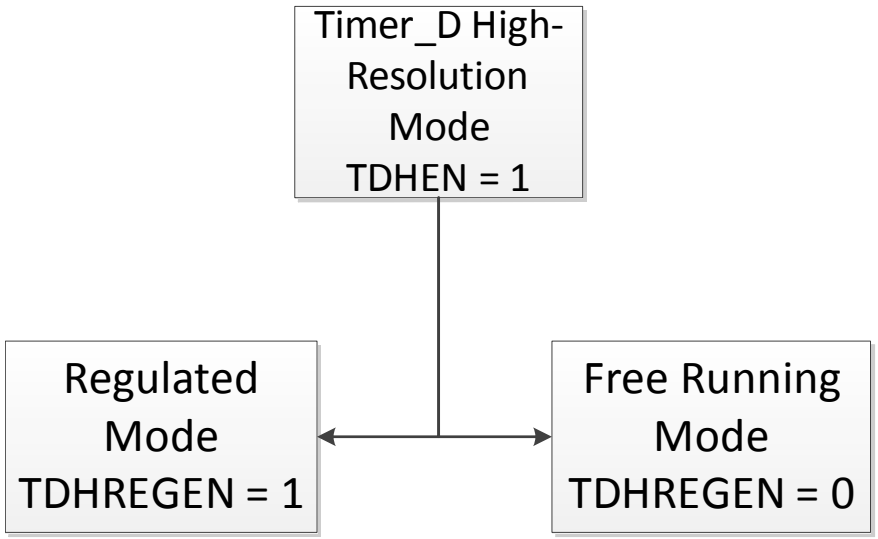

Figure 2. Timer_D High-Resolution Operation Modes

3 Free-Running Mode

When TDHREGEN = 0, Timer_D operates in free-running mode

3.1 Timer Source Clock

In free-running mode, the Timer_D clock source selected by TDSSELx is ignored. Instead, a Timer_D internal local clock generator is used as the source for the Timer_D hi-res clock generator. This clock generator is internal to the high-resolution generator as shown in Figure 3 and does not require any input clock source external to the Timer_D module to operate. Hence, this oscillator enables the operation of Timer_D when no other clock source, such as SMCLK, is available. As a result, Timer_D is operational in LPM0 through LPM4 without the need for an external clock. The current consumption in LPMx is dominated by the Timer_D power specifications; hence, currents higher than the ones specified for the low-power modes are observed when Timer_D is running. The overview of the high-resolution generator is shown in Figure 3.

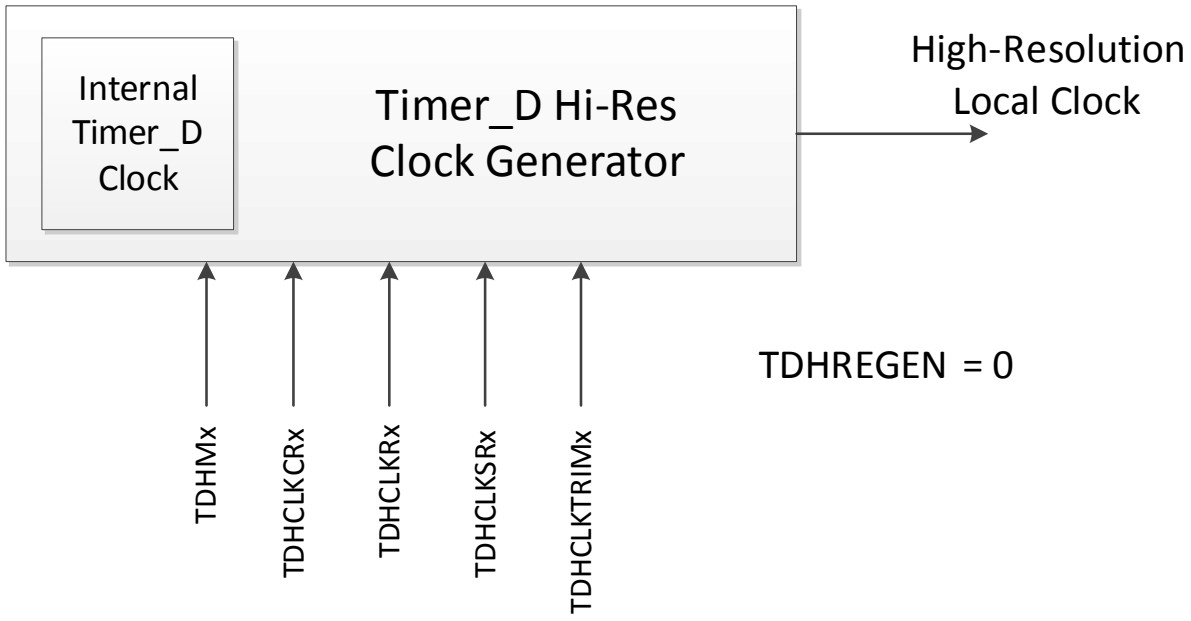

Figure 3. Free-Running Mode High-Resolution Block

NOTE: It is important to distinguish the internal Timer_D Clock from the high-resolution local clock. The internal Timer_D clock is an input to the high-resolution generator. The high-resolution local clock is an output of the high-resolution generator.

3.2 Frequency Adjustments

When operating in free-running mode, the frequency of the timer is determined by the bits shown in Table 2. These bits are configured in software during run time.

Table 2. Frequency Adjustment Bits for Free-Running Mode

Bits	Register Name	Description
TDHMx	TDxHCTL0	Timer_D high-resolution clock multiplication factor.
TDHCLKCR	TDxHCTL1	Timer_D high-resolution coarse clock range selection bit. Must be set for a source clock > 15 MHz.
TDHCLKRx	TDxHCTL1	Bits used to define the coarse clock range of the high-resolution clock generator.
TDHCLKSRx	TDxHCTL1	Timer_D high-resolution clock sub-range selection bits. About ±3% frequency adjustment for each step of TDHCLKSRx
TDHCLKTRIMx	TDxHCTL1	Timer_D high-resolution clock trim selection bits. About ±1.5% frequency adjustment for each step of TDHCLKTRIMx

In free-running mode, the frequency of Timer_D for a respective bit configuration is given in the “Timer_D, Local Clock Generator Frequency” electrical characteristics table of the device-specific data sheet. Table 3 shows an example of a snippet of the table for MSP430F5172. For frequency information, see the device-specific data sheet. The programmer must ensure that the configuration bit settings selected for the high resolution generator is ≤ 256 MHz. Any frequency above 256 MHz can be functional, but it is outside of the device specifications.

Table 3. Table Example for Local Clock Generator Frequency (over recommended ranges of supply voltage and operating free-air temperature – unless otherwise noted)

Parameter		Test Conditions	Min	Typ	Max	Unit
$f_{HRCG(0,0,64)}$	HRCG frequency (0, 0, 64)	TDHREGEN = 0, TDHMx = 0, TDHCLKCR = 1, TDHCLKRx = 0, TDHCLKSRx = 0, TDHCLKTRIM = 64	39	56	73	MHz
		TDHREGEN = 0, TDHMx = 1, TDHCLKCR = 1, TDHCLKRx = 0, TDHCLKSRx = 0, TDHCLKTRIM = 64	78	112	146	
$f_{HRCG(0,7,64)}$	HRCG frequency (0, 7, 64)	TDHREGEN = 0, TDHMx = 0, TDHCLKCR = 1, TDHCLKRx = 0, TDHCLKSRx = 7, TDHCLKTRIM = 64	46	66	86	MHz
		TDHREGEN = 0, TDHMx = 1, TDHCLKCR = 1, TDHCLKRx = 0, TDHCLKSRx = 7, TDHCLKTRIM = 64	92	132	172	

Note that for each bit combination in Table 3, a wide range of frequency is presented. For instance, if the application needs Timer_D to run at 56 MHz, then the first configuration from Table 3 is a good starting point since it shows a typical frequency of 56 MHz; however the actual frequency of the timer will be anywhere between 39 MHz and 73 MHz. This accuracy may not be acceptable in some applications, and it is a result of variations on operating voltage, temperature and variations in the process of the silicon. Thus, calibration is required to achieve the desired frequency in free-running mode. Section 3.3 describes a technique to calibrate the timer to overcome this problem.

3.3 Frequency Calibration Technique for Timer_D

This section presents one technique for calibrating Timer_D to a desired frequency. Given its temperature and voltage dependence and device variations, calibration is required in most applications using Timer_D in free-running mode.

First, it is important to understand the steps required to change the frequency once the timer is running. These steps are presented in the *MSP430x5xx and MSP430x6xx Family User's Guide* (SLAU208) and repeated here for convenience:

1. Increment TDHCLKTRIMx by 1 until TDHCLKTRIMx = 64 is reached. If the clock range must be changed, proceed to Step 1a for a higher clock range or to Step 1b for a lower clock range.
   (a) If the clock range, TDHCLKRx, must be changed to a higher clock range, then the TDHCLKSRx must be brought to TDHCLKSRx = 31 by incrementing by 1.
   (b) If the clock range, TDHCLKRx, must be changed to a lower clock range, then the TDHCLKSRx must be brought to TDHCLKSRx = 0 by decrementing by 1.
2. Change the clock range, TDHCLKRx, after step 1a or 1b. Increment or decrement by 1 at a time.
3. Increment or decrement TDHCLKSRx by 1 until the desired clock frequency is reached to an accuracy of approximately ±3%.
4. Increment or decrement TDHCLKTRIMx by 1 until the desired clock frequency is reached to an accuracy of approximately ±1.5%.
5. Further changes to track the frequency because of changes in the environment can be typically done by changing only the TDHCLTRIMx values.

In order to calibrate the timer at run time, feedback from the timer output is required. One way to obtain this required feedback is to use a slower clock, such as ACLK running at f_{SLOW} Hz, and the input capture feature of Timer_D. By capturing the edges of the slow timer running at a known frequency, and obtaining the difference (TDR_2- TDR_1) in Timer_D counts between slower timer edges as shown in Figure 4, the actual frequency of Timer_D can be determined; hence, providing system feedback for calibration. Note that the accuracy of calibration will be given by the accuracy of f_{SLOW}. Since f_{SLOW} is known, the expected Timer_D count difference for the elapsed time between ACLK edges can be calculated; consequently, evaluating the actual number of TDR_2- TDR_1, and adjusting TDxHCTL1 until the expected count value is achieved results in a calibrated timer.

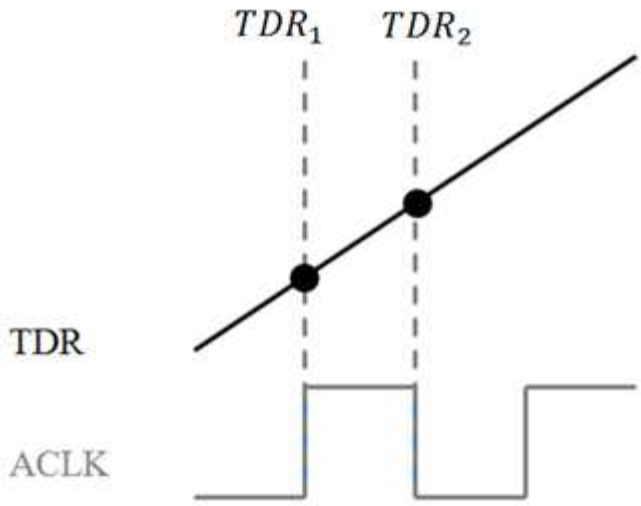

Figure 4. Timer_D Calibration Using ACLK

Timer_D is initialized to the desired range shown in Table 3. In this example, ACLK running at 32768Hz is then sourced to the input capture pin of Timer_D. In the MSP430F5172, this connection can be made in software without the use of any external pins, since TD0.2 is internally connected through a multiplexer (see Figure 5) to ACLK when the input CCI2B is selected as shown in Table 4. For Timer_D signal connections, see the device-specific data-sheet.

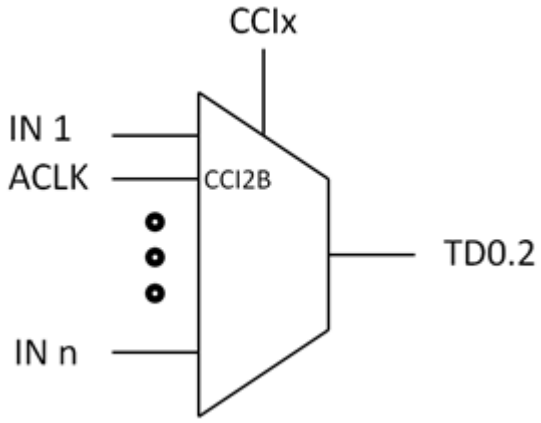

Figure 5. CCIx Multiplexer for Signal Connections

Table 4. Internal connection of ACLK to TD0.2 With CCI2B Selection

Input Pin Number	Device Input Signal	Module Input Signal
ACLK (Internal)	TD0.2	CCI2B

Timer_D is initialized to run in continuous mode to minimize the probability of getting a timer overflow during calibration. Timer_D counts from 0 to 0xFFFF, or 65535_{10}, in $t_{overflow}$ seconds:

$$t_{overflow} = \frac{1}{f_{desired}} * 65535 \bullet \text{seconds} \quad (1)$$

Where $f_{desired}$ is the desired final frequency of Timer_D. Figure 6 shows Timer_D counting in increments of $\frac{1}{f_{desired}}$.

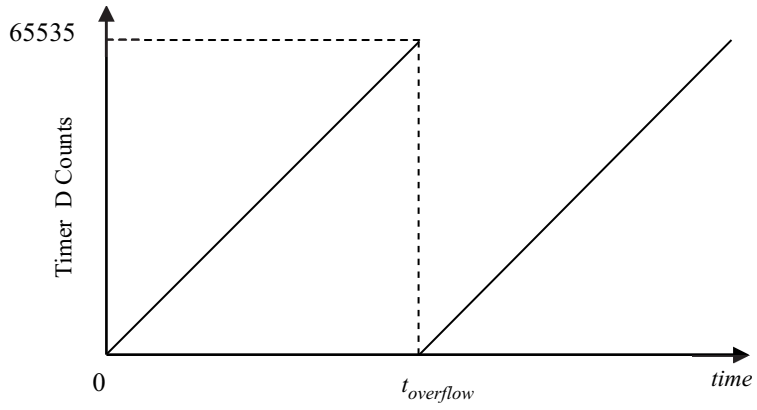

Figure 6. Counts vs. t_{TIMERD} for Timer D. Each Count Takes $\frac{1}{f_{desired}}$ Seconds

The input capture feature of Timer_D is selected to trigger an interrupt on both edges of ACLK. The time between edges, $t_{ACLK_{1/2}}$, is 1/2 the period of ACLK as shown in Figure 7.

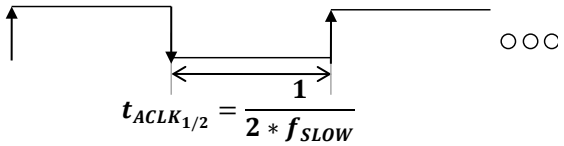

Figure 7. ACLK Timing Waveform

The expected Timer_D counts between ACLK edges, *Expected Count*$_{TIMERD}$, can be calculated from $f_{desired}$ and t_{ACLK}:

$$Expected\,Count_{TIMERD} = t_{ACLK_{1/2}} * f_{desired} \quad (2)$$

Similarly, the actual number of counts between ACLK edges is:

$$Actual\,Count_{TIMERD} = t_{ACLK_{1/2}} * f_{actual} \quad (3)$$

Actual *Count*$_{TIMERD}$ is proportional to the output frequency of the timer; hence, it provides feedback to the system and can be used for calibration. If the actual frequency of Timer_D, f_{actual}, is higher than $f_{desired}$, then *Actual Count*$_{TIMERD}$ will be higher than *Expected Count*$_{TIMERD}$ and the Timer_D frequency is lowered by adjusting the TDHCLKTRIMx bits down. Similarly, if the actual frequency of Timer_D, f_{actual}, is lower than $f_{desired}$, then *Actual Count*$_{TIMERD}$ will be lower than *Expected Count*$_{TIMERD}$ and the Timer_D frequency is increased by adjusting the TDHCLKTRIMx bits up.

In this example, the interrupt service routine of Timer_D, TIMER0_D1_VECTOR, contains the TDCCR2 CCIFG vector, which is serviced at every captured edge of f_{SLOW}. When a capture is performed the following occurs in the presented order:

- The timer value is copied from TDR into the Timer_D Capture/Compare Register (TDCCRx).
- The TDCCRx register is copied into the Timer_D Compare Latch Register (TDCLx).
- The interrupt flag (CCIFG) is set.
- The Timer Capture Overflow bit is set in case of a timer capture overflow condition.

Hence, the register TD0CL2 contains the count value of Timer_D at the capture event. If the value of TD0CL2 is *TD0CL2*$_1$ at event capture number N, and the value of TD0CL2 is *TD0CL2*$_2$ at event capture number *N+1*, then:

$$Actual\,Count_{TIMERD} = TD0CL2_2 - TD0CL2_1 \quad (4)$$

The goal is to adjust TDHCLKTRIMx until *Actual Count*$_{TIMERD}$= *Expected Count*$_{TIMERD}$. Care must be taken or this condition may result in an infinite loop. An infinite event can occur when increasing TDHCLKTRIMx by 1 count makes *Actual Count*$_{TIMERD}$ > *Expected Count*$_{TIMERD}$ and decreasing TDHCLKTRIMx by 1 count makes *Actual Count*$_{TIMERD}$ < *Expected Count*$_{TIMERD}$. This condition is possible because adjusting TDHCLKTRIMx changes the timer frequency by approximately 1.5%. To prevent this problem a tolerance around *Expected Count*$_{TIMERD}$ should be allowed:

Expected Count$_{TIMERD}$= *Expected Count*$_{TIMERD}$ ± *Range* (prevent infinite calibration loop)

The allowed range depends on the accuracy required and varies for all applications. The best accuracy that can be achieved shows a minimum value of 1.5%, since each step of TDHCLKTRIMx changes the frequency by 1.5%. The accuracy is a multiple of 1.5%. For instance, if the required accuracy is 2%, then the range must be set for 1.5%. If the required accuracy is 3.8%, then the range must be set for an accuracy of 3%. Figure 8 shows the software flowchart for calibrating Timer_D. An example is presented in Table 5.

The frequency required by the timer calibration routine depends strictly on the end application and its environmental conditions. For instance, an application expected to undergo rapid changes of temperature should calibrate more often than a system located in a more stable environment.

NOTE: System clock must be fast enough to allow enough time between capture interrupts to enter, execute and exit the Timer_D ISR. The required time may depend on several factors including compiler, optimization settings, and so forth.

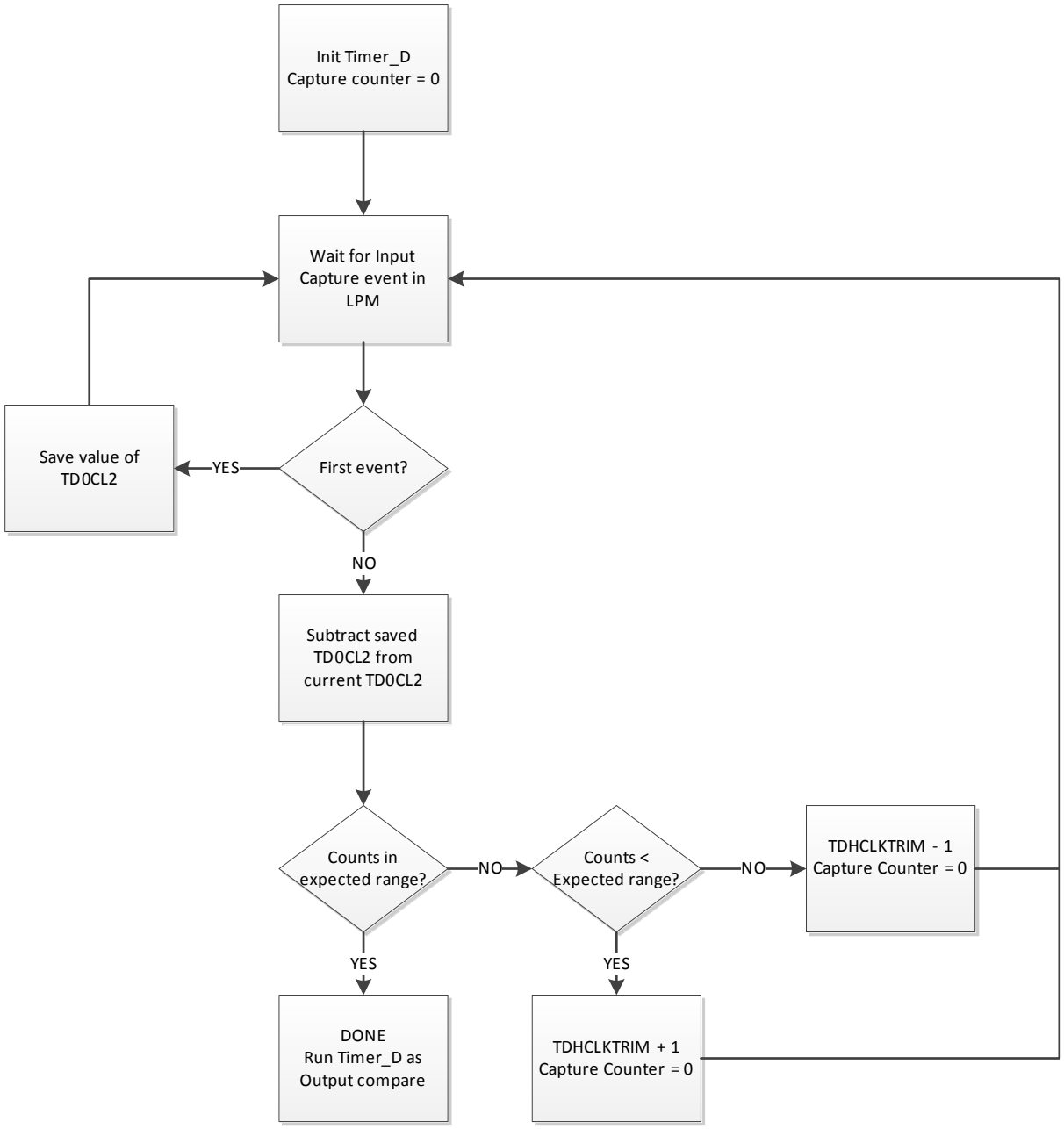

Figure 8. Software Flowchart for Timer_D Calibration

3.4 Tag-Length-Value (TLV) Calibration Constants

There are factory pre-programmed values stored in the TLV structure of the Flash memory for the TDHCLKRx, TDHSRx, and TDHCLKTRIMx bits of TDHCTL1 for specific frequencies in free-running mode. To use the calibrated settings, the register setting is copied into the corresponding registers. In addition to these settings, a clock multiplication factor and coarse clock range selection must be applied to the TDHMx and TDHCLKCR bits of the TDxHCTL1 register. Frequency calibration is currently provided for 64 MHz, 128 MHz, 200 MHz and 256 MHz. Table 5 shows an example of a table from an MSP430F51xx device as an example. For more information on the calibration constants, see the device-specific data sheet.

Table 5. Timer_D Trimmed Clock Frequencies Sample (over recommended ranges of supply voltage and operating free-air temperature – unless otherwise noted)

Parameter		Test Conditions	Min	Typ	Max	Unit
	Frequency tolerance during trimming		-0.5		+0.5	%
$f_{TRIM(64MHz)}$	TDHMx = 0, TDHREGEN = 0, TDHCLKCR = 0, TDHxCTL1 = TDHxCTL1_64	T_A = 25°C, V_{CC} = 1.8 V	63	64	65	MHz
$f_{TRIM(128MHz)}$	TDHMx = 0, TDHREGEN = 0, TDHCLKCR = 1, TDHxCTL1 = TDHxCTL1_128	T_A = 25°C, V_{CC} = 2.0 V	126	128	130	MHz
$f_{TRIM(200MHz)}$	TDHMx = 0, TDHREGEN = 0, TDHCLKCR = 1, TDHxCTL1 = TDHxCTL1_200	T_A = 25°C, V_{CC} = 2.4 V	197	200	203	MHz
$f_{TRIM(256MHz)}$	TDHMx = 1, TDHREGEN = 0, TDHCLKCR = 1, TDHxCTL1 = TDHxCTL1_256	T_A = 25°C, V_{CC} = 2.2 V	250	256	262	MHz

A high-resolution clock generator (HRCG) voltage and temperature drift is provided in the device-specific data sheet. A sample of the table is shown in Table 6. For MSP430F51xx, see the device-specific data sheet.

Table 6. HRCG Voltage and Temperature Drift Sample Table (over recommended ranges of supply voltage and operating free-air temperature – unless otherwise noted)

Parameter		Test Conditions	Min	Typ	Max	Unit
df_{HRCG}/dT	HRCG frequency temperature drift	f_{HRCG} = 8 MHz, TDHREGEN = 0			±0.17	%/°C
		f_{HRCG} = 16 MHz, TDHREGEN = 0			±0.16	%/°C
		f_{HRCG} = 25 MHz, TDHREGEN = 0			±0.16	%/°C
		f_{HRCG} = 8/16/25 MHz, TDHREGEN = 1		0		%/°C
$df_{HRCG}/dVCC$	HRCG frequency voltage drift	f_{HRCG} = 8/16/25 MHz, TDHREGEN = 0	0		5	%/°V
		f_{HRCG} = 8/16/25 MHz, TDHREGEN = 1		0		%/°V

The tolerance of the values stored in the TLV structure for Timer_D is given by a combination of the values shown in Table 5 and Table 6. For instance, when using the factory calibrated frequency for 256 MHz on this particular device at 3.2 V is given by:

Calibration Constant Range = Range at 2.2 V + 5% = 250 to 262 MHz ±5% = 237 to 275 MHz

since there is a 5% frequency drift per volt. Note that the above range does not include temperature variations.

For some applications, this tolerance is not acceptable and the technique for auto-calibration should be employed. The TLV constants may be used to initialize Timer_D to the target frequency and then the calibration routine may be run to obtain the desired frequency.

3.5 Software Snippet for Calibrating Timer_D

To summarize, Timer_D is configured as a 16-bit timer sourced from the Internal Timer_D Clock, high resolution, enhanced accuracy, in continuous mode and with input capture functionality with interrupts enabled on the rising and falling edges of ACLK routed internally to TD0.2. The TLV constant for 256 MHz is used as the initial value of the TD0HCTL1 register. The initialization of Timer_D for calibration is shown below. The addresses of TLV constants are given in the device-specific data sheet. In this example, the constant for 256 MHz is stored at address 0x1A36.

```
#define CALTDH0CTL1_256      *((unsigned int *)0x1A36)

    TD0HCTL1 = CALTDH0CTL1_256;  // Get the 256 Mhz TimerD TLV Data
    TD0CTL0 = CNTL_0             // 16-bit timer
             + ID_0              // Divide by 1
             + MC_0;             // Halt timer until init is complete

    TD0CTL1 |= TDCLKM_1;         // TD0 clock = Hi-res local clock

// FREE RUNNING MODE (TDHREGEN = 0)
    TD0HCTL0 = TDHFW             // Fast wake-up mode.
             + TDHD_0            // Set divider to 1
             + TDHM_1            // Multiply by 16
             + TDHEAEN           // Enhance accuracy
             + TDHEN;            // Enable Hi-Res

    TD0CTL2 &= ~TDCAPM2;         // Channel 2 single capture mode

    TD0CCTL2 = CAP               // Capture mode
             + CM_1              // Rising and falling edge
             + CCIE              // Enable local interrupt for Timer TD0.2
             + CCIS_1;           // Use ACLK running at 32768 Hz as input
```

Code example with the calibration routine is provided with this document at http://www.ti.com/lit/zip/slaa601.

3.6 Frequency Calibration Technique for Timer_D: An Example

This section presents an example for calibrating Timer_D. Given the following application requirements:

$$f_{desired} = 256\ MHz$$
$$f_{SLOW} = 32768\ Hz\ (from\ ACLK)$$
$$Accuracy = f_{desired}\ \pm 2\% \quad (5)$$

Then,

$$t_{ACLK} = \frac{1}{2 * f_{SLOW}} = \frac{1}{2 * 32768} = 0.015258789\ ms$$
$$Expected\ Count_{TIMERD} = t_{ACLK} * f_{desired} = 0.015258789\ ms * 256\ MHz = 3906$$
$$Range = Expected\ Count_{TIMERD}\ \pm 1.5\% = 3906 \pm 58 \quad (6)$$

Note that even though the application requires an accuracy of 2.2%, the range must be calculated for 1.5%. The next accuracy steps would be 3%, 4.5%, 6% and so on. 58 is the total number of counts that will ensure the frequency to be within 1.5% of the desired frequency

3.7 Frequency Calibration Accuracy

The presented calibration technique presents two sources of error:

- Tolerance of f_{SLOW}
- ±1.5% of the desired frequency given by each step of TDHCLKTRIMx.

The tolerance of f_{SLOW} is given by the tolerance of the source clock. For instance, if ACLK is used and sourced from REFO, the tolerance is ±3.5% across the full voltage and temperature range as specified in the MSP430F51xx data sheet. Using an external low-frequency crystal yields a lower tolerance and a better Timer_D clock accuracy.

3.8 Frequency Calibration Accuracy Lab Test

The robustness of the self-calibration firmware over varying temperatures and voltages was tested in the lab. Testing was done at 85°C, 20°C, and -40°C for the device running on a 3.0 V and 2.5 V power supply. Seven MSP430 devices were programmed to auto-calibrate Timer_D to 256 MHz. It has been observed that frequency is proportional to temperature. A dampening effect is also observed on the frequency when the voltage of the power supply driving the device is decreased. The following data and test procedures observe the effectiveness of the self-calibration system. A soaking time of 7 minutes was used to ensure temperature equilibrium before measurements were taken.

3.8.1 Equipment

- MSP-TS430RSB40 target board with programmed MSP430F5172
- Thermostream and proper insulation materials for testing
- Power supply used to vary voltage driving the device
- Alligator clips or wires
- Oscilloscope probes for frequency reading

3.8.2 Connections

- Timer D output to Oscilloscope
- V_{CC} and GND from power supply

Figure 9 shows the test flowchart for the Timer D calibration.

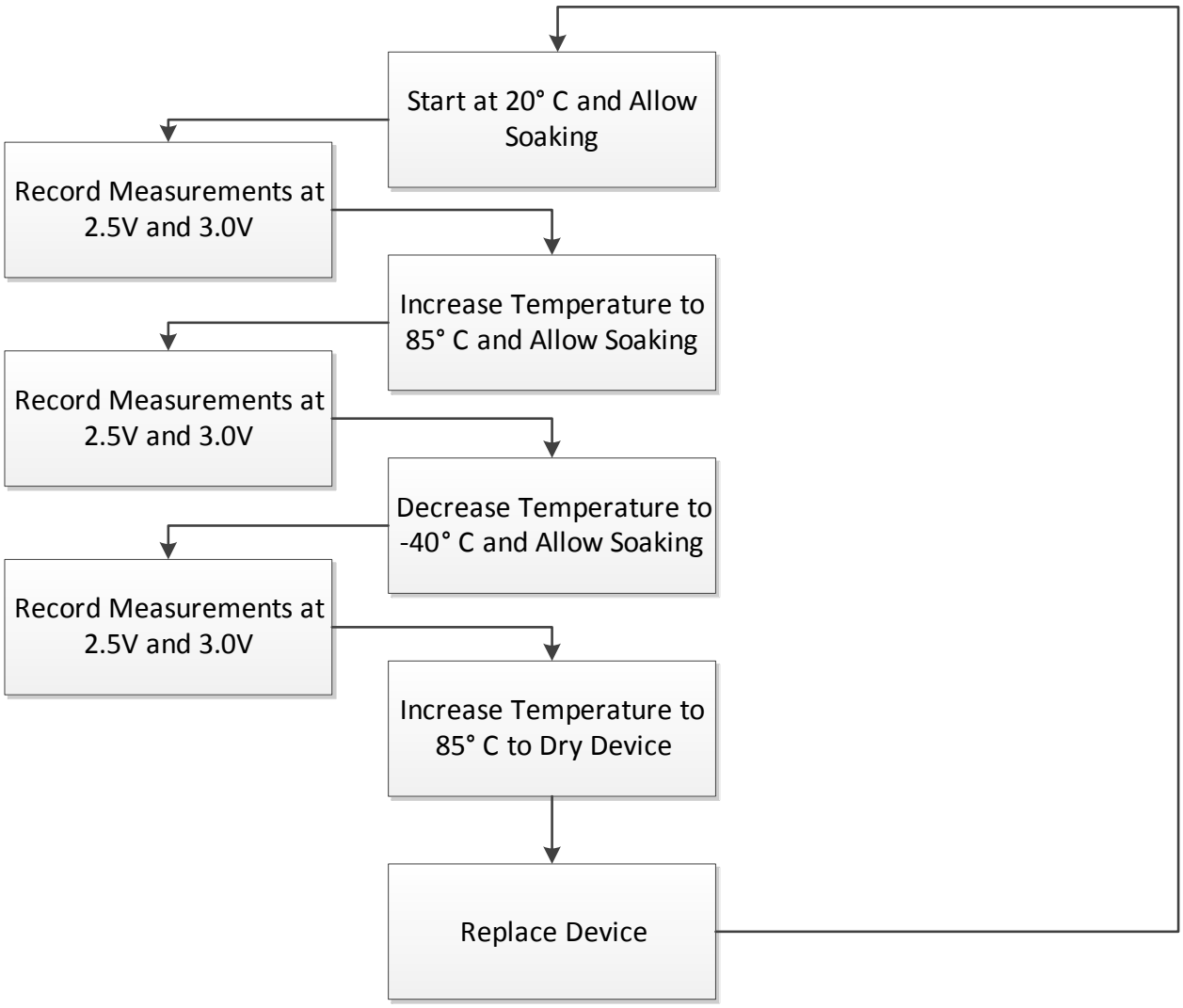

Figure 9. Timer_D Calibration Test Flowchart

Collected data was then analyzed. The deviation from baseline (256 MHz) was found and shown in Figure 10.

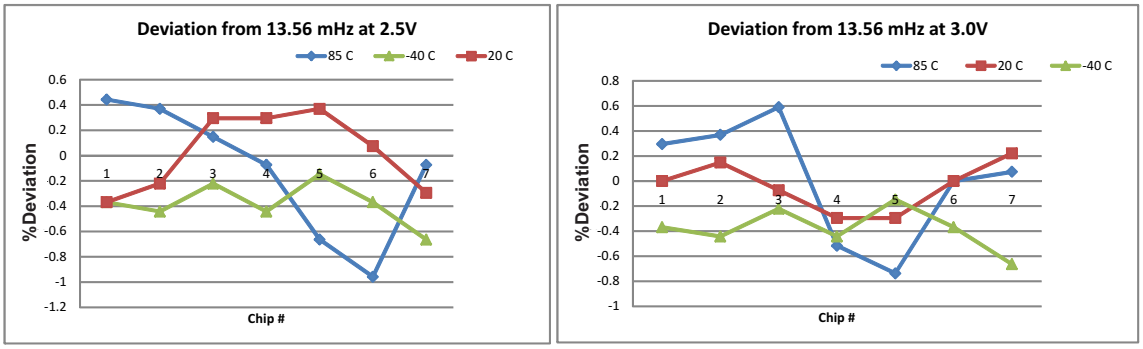

Figure 10. Test Results With calibration ON

Results show a percentage variation of less than 1% from 256 MHz over the temperature and voltage operating range of the MSP430F51x2. Calibration was performed at a reset event. In the test setup, a reset was issued after the temperature and voltage was varied.

For comparison purposes, Figure 11 show the data collected at 2.5 V and 3 V, respectively, for a temperature sweep without the Timer_D calibration routine.

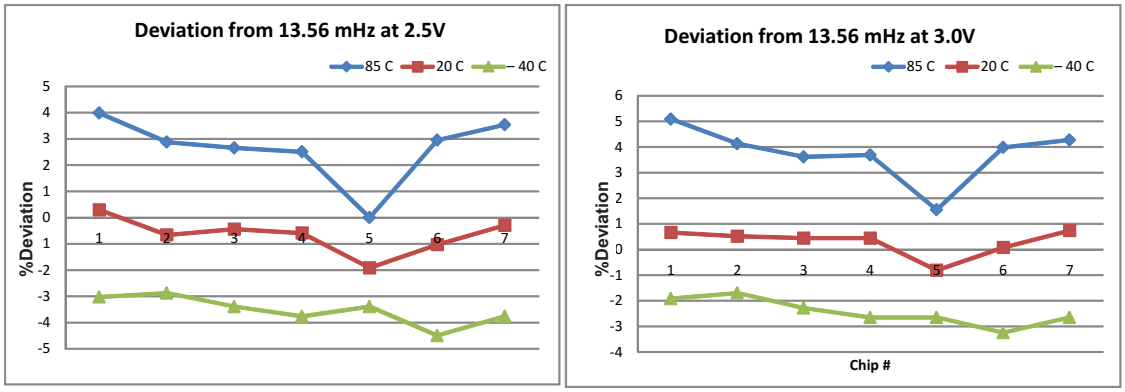

Figure 11. Test Results With calibration OFF

Results show a percentage variation of over 5% from 256 MHz over the temperature and voltage operating range of the MSP430F51x2 with the Timer_D calibration set to OFF.

4 Regulated Mode

When TDDREGEN = 1, Timer_D operates in regulated mode.

4.1 Timer Source Clock

In regulated mode, the Timer_D clock source is selected by TDSSELx. The input clock is fed to the high-resolution generator and its output sources the 16-bit timer TDR register through the TDCLKMx multiplexer as shown in Figure 12.

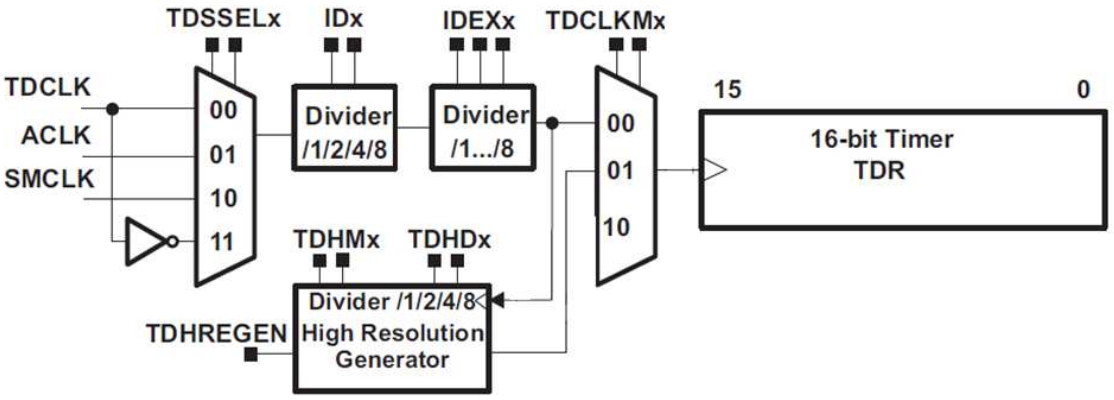

Figure 12. Regulated Mode High-Resolution Block

Note that whether Timer D is in free-running or regulated mode, TDCLKMx must be selected as the high-resolution local clock (TDCLKMx = 01b). Programmers may be incorrectly inclined to select the external clock source for TDCLKMx, since an external clock is used to generate the high-resolution clock of the Timer_D module. The external source clock sources the high-resolution generator, but the Timer_D clocking mode selector is the high-resolution local clock just as in the case of free-running mode; otherwise, the high resolution generator is bypassed.

4.2 Frequency Adjustments

When operating in regulated mode, the frequency of the timer is determined by the frequency of the source clock and the timer configuration bit setting in TDHCLKCR, TDHMx and TDHDx. In contrast to the free-running mode, TDHCLKRx, TDHCLKSRx, and TDHCLKTRIMx are modified automatically by hardware in the regulated mode. Timer_D accuracy is controlled by the accuracy of the input source clock, and no auto-calibration is required. The programmer must ensure that the frequency generated by the high-resolution generator is ≤ 256 MHz. For instance, for a 25 MHz source clock, a multiplier of 16 is not valid, since it drives the timer to operate outside of its operation specifications and its operation is not ensured.

4.3 Software Snippet for Initializing Timer_D in Regulated Mode

The following code snippet shows how to initialize Timer_D in high-resolution, regulated mode. An input clock of 16 MHz is assumed on SMCLK.

```
TD0CTL0 = TDSSEL_2     // Use SMCLK – 16 MHz assumed
        + CNTL_0       // 16-bit timer
        + ID_0         // Divide by 1
        + MC_0;        // Halt timer until init is complete

TD0CTL1  |= TDCLKM_1;  // TD0 clock = Hi-res local clock

TD0HCTL0 = TDHFW       // Fast wake-up mode.
         + TDHD_0      // Set divider to 1
         + TDHM_1      // Multiply by 16. 16 MHz x 16 = 256 MHz
         + TDHREGEN    // Regulated Mode
         + TDHEAEN     // Enhance accuracy
         + TDHEN;      // Enable Hi-Res
```

5 TIMER_D High-Resolution PWM Generation

When generating a pulse-width modulated signal in high-resolution mode, some special considerations must be kept in mind. This section describes these considerations as well the limitations in PWM generation of the Timer_D module.

A PWM signal is described by two parameters: frequency and duty cycle. The frequency of the PWM is determined by the following components:

- Timer Frequency
- Value in TDxCCR0
- Timer Length: 8, 10, 12 or 16 bits given by CNTLx bits
- Timer Mode: Up, down, continuous or up/down given by MCx bits

The PWM duty cycle is controlled by the value of TDxCCRn. For basics on PWM generation, see *Timer_D Module (Chapter Excerpt From MSP430x5xx and MSP430x6xx Family User's Guide, SLAU208)* (SLUA323).

NOTE: UP/DOWN mode cannot be used in hi-resolution mode. Unexpected behavior is observed.

5.1 PWM Period Resolution

One of the most important characteristics to be considered when using the compare latch TDxCL0 as a period register under high-resolution mode is that the four LSB of the TDxCL0 in the 16x case (TDHM = 1) or the three LSB in the 8x case (TDHM = 0) are ignored and replaced by 1s. Table 7 shows effective TDxCL0 values for various TDxCL0 configurations. The "TDxCLO value" is the actual number loaded to the TDxCL0 register. The "effective TDxCL0" value is the number the hardware evaluates for the frequency of the PWM signal.

Table 7. Effective TDxCL0 Values for Various TDxCL0 Configurations (1)

TDxCL0 Value (decimal)	TDxCL0 Value (binary)	Multiplier	Effective TDxCL0 Value (binary)	Effective TDxCL0 Value
1	0**0001**	x16	0**1111**	15
	00**001**	x8	00**111**	7
9	0**1000**	x16	0**1111**	15
	01**000**	x8	01**111**	15
15	0**1111**	x16	0**1111**	15
	01**111**	x8	01**111**	15
17	1**0001**	x16	1**1111**	31
	10**001**	x8	10**111**	23
26	1**1010**	x16	1**1111**	31
	11**010**	x8	11**111**	31

(1) Bits highlighted in bold are ignored and replaced by 1s.

What does this mean in terms of PWM period resolution? The PWM period event is indicated when the TDIFG flag is set. TDIFG is set at different events for each of the timer mode control:

- Up Mode: TDIFG is set when the timer counts from TDxCL0 to 0
- Continuous Mode: TDIFG is set when the timer counts from TDR_{max} to 0
- Up/Down Mode: TDIFG is set when the timer counts down from 1 to 0

Thus, the period of a PWM signal depends on the timer mode control:

- Up Mode:

$$PWM_{period_{up}} = \frac{2 * (TDxCL0_{effective} + 1)}{Frequency_{TIMERD}} \text{ (Toggle Mode)}$$

$$PWM_{period_{up}} = \frac{TDxCL0_{effective} + 1}{Frequency_{TIMERD}} \text{ (Set/Reset and Reset/Set Modes)} \quad (7)$$

- Continuous Mode:

$$PWM_{period_{cont}} = \frac{2 * (TDR_{max} + 1)}{Frequency_{TIMERD}} \text{ (Toggle Mode)}$$

$$PWM_{period_{cont}} = \frac{TDR_{max} + 1}{Frequency_{TIMERD}} \text{ (Set/Reset and Reset/Set Modes)} \quad (8)$$

- Up/Down:

$$PWM_{period_{cont}} = \frac{4 * (TDR_{max} + 1)}{Frequency_{TIMERD}} \text{ (Toggle Mode)}$$

$$PWM_{period_{cont}} = \frac{2 * (TDR_{max} + 1)}{Frequency_{TIMERD}} \text{ (Set/Reset and Reset/Set Modes)} \quad (9)$$

In the high-resolution regulated mode with 16x multiplier selected, the 4 LSB of TDxCL0 are ignored, thus each period step is every 2^4 counts of TDxCL0. Similarly, in the high-resolution regulated mode with 8x multiplier selected, each period step is every 2^3 counts of TDxCL0. Since the ignored bits are replaced by 1s, a factor of 16 and a factor of 8 is added to the TDHMx = 1 case and the TDHMx = 0 case respectively. For up mode in a non-toggle output configuration, the PWM period granularity that can be achieved is:

$$Steps\_PWM_{period_{x16}} = \frac{TDxCL0_{effective} + 1}{Frequency_{TIMERD}} = \frac{16 + n * 2^4}{Frequency_{TIMERD}}$$

$$Steps\_PWM_{period_{x8}} = \frac{TDxCL0_{effective} + 1}{Frequency_{TIMERD}} = \frac{8 + n * 2^3}{Frequency_{TIMERD}}$$

$$where\ n\ is\ an\ integer > 0 \quad (10)$$

To illustrate this behavior, consider the following example:

Given,

$$Frequency_{TIMERD} = 256\ MHz$$

$$TDxCL0 = 0\ to\ 64 \quad (11)$$

For the x16 case, there are $\frac{64 - 0}{2^4} - 1 = 3$ non-zero steps for TDxCL0 from 0 to 64. Similarly, for the x8 case, there are $\frac{64 - 0}{2^3} - 1 = 7$ non-zero steps for TDxCL0 from 0 to 64. The -1 factor comes from the fact that the first step for TDxCL0 from 0 to 15 in the 16x case and for TDxCL0 from 0 to 7, results in either 0% or 100% (0Hz effective frequency) duty cycle due to high-resolution mode limitations. For more information, see Section 5.3.

The non-zero PWM period steps are:

$$Steps_{PWM_{period_{x16}}} = \left(\frac{16 + n * 2^4}{256\ x\ 10^6}\right)^{for\ n = 1,2,3}$$

$$= 125\ ns, 187.5\ ns\ and\ 250\ ns$$

$$= 8\ MHz, 5.33\ MHz, 4\ MHz \quad (12)$$

$$Steps_{PWM_{period_{x8}}} = \left(\frac{8 + n * 2^3}{256\ x\ 10^6}\right)^{for\ n = 1, 2, 3, 4, 5, 6, 7}$$

$$= 62.5\ ns, 93.75\ ns, 125\ ns, 156.25\ ns, 187.5\ ns, 218.75\ ns\ and\ 250\ ns$$

$$= 16\ MHz, 10.6\ MHz, 8\ MHz, 6.4\ MHz, 5.33\ MHz, 4.57\ MHz\ and\ 4\ MHz \quad (13)$$

In the regulated mode, software does not control the TDHCLKRx, TDHCLKSRx, or TDHCLKTRIMx and the frequency resolution is strictly limited by TDxCL0 for a fixed source clock as shown in the above example.

In free-running mode, software control of TDHCLKTRIMx allows the adjustment of $Frequency_{TIMERD}$ by ±1.5%, which in turn results in a PWM_{period} resolution of 1.5% as shown in the following example.

Given,

$$Frequency_{TIMERD} = 256\ MHz$$

$$TDxCL0 = 27$$

$$TDHM = 1$$

$$Timer\ Mode = Up \quad (14)$$

Since TDHM = 1, the 4 LSB of TDxCL0 are ignored and replaced by 1s. So 27_{10}= 11011_2 results in an effective TDxCL0 value of 31_{10} = 11111_2.

For this example, the timer is configured in up mode, with an effective TDxCL0 value of 31. TDIFG is set when the timer counts from 31 to 0, so the PWM frequency is given by:

$$PWM_{period} = \frac{TDxCL0_{effective} + 1}{Frequency_{TIMERD}} - \frac{31+1}{256\ x\ 10^6} = 8MHz \quad (15)$$

Adjusting $Frequency_{TIMERD}$ by -1.5%:

$$Frequency_{TIMERD\_adjusted} = 256\ MHz - 256\ MHz * 1.5\% = 252.16\ MHz \quad (16)$$

and,

$$PWM_{period\_adjusted} = \frac{TDxCL0_{effective} + 1}{Frequency_{TIMERD}} \frac{31+1}{252.16\ x\ 10^6} = 7.88\ MHz \quad (17)$$

Therefore, in free-running mode a higher PWM period resolution can be achieved in comparison to regulated mode.

5.2 PWM Frequency Limits

This section discusses the maximum and minimum achievable PWM frequencies with Timer_D in high-resolution mode.

The minimum PWM frequency is achieved in 16-bit length, up/down mode with a toggled output and it is given by:

$$PWM_{frequency_{MIN}} = \frac{\text{Frequency}_{\text{TIMERD}}}{4 * (0xFFFF + 1)} \quad (18)$$

The maximum PWM frequency is limited by the following parameters:

- High-resolution mode (free-running or regulated)
- Required duty cycle of the signal
- Chosen output mode (OUTMODx).

Duty cycle limitations can be found in the *MSP430x5xx and MSP430x6xx Family User's Guide* (SLAU208). To illustrate calculations of maximum PWM frequency, an example is provided with this document.

Consider the following requirement:

$$Duty\ Cycle = 50\%$$
$$OUTMODE = Any$$
$$Hi-resMode = Regulated\ with\ SMCLK\ 16\ MHz \quad (19)$$

The maximum PWM frequency is achieved when the frequency of Timer_D is at its maximum (256 MHz), and it is limited by the following illegal conditions:

$$TDHMx = 0 : TDxCLn \le 0x0007$$
$$TDHMx = 1 : TDxCLn \le 0x000F$$
$$TDxCLn \ge TDxCL0 - 0x0008 \quad (20)$$

For 50% duty cycle, $TDxCLn = \frac{1}{2}(TDxCL0_{effective} + 1)$. With a 16 MHz SMCLK as a source clock to the system, TDHMx must be 1 to obtain a Timer_D frequency of 256 MHz. The smaller the value of TDxCL0, the higher the frequency of the PWM signals, so the goal is to make TDxCL0 as small as possible. The low limit for Timer_D Capture/Compare Latch Register (TDxCLn) is given by TDxCLn ≤ 0x000F since TDHMx is 1. TDxCL0 must be twice the value of TDxCLn – 1 for a 50% duty cycle.

Thus,

$$TDxCLn_{MIN} = 16$$
$$TDxCL0_{effective} = 2 * TDxCLn_{MIN} - 1 = 31 \quad (21)$$

Up mode provides the highest PWM frequency, so:

$$PWM_{frequency_{MAX}} = \frac{\text{Frequency}_{\text{TIMERD}}}{TDxCL0_{effective_{MIN}} + 1} = \frac{256\ x\ 10^6}{32} = 8\ \text{MHz}\ (\text{for}\ \ TDHM = 1) \quad (22)$$

Now, the same example is repeated for:

Hi-res Mode = Regulated with SMCLK = 25 MHz

For an input clock of 25 MHz, TDHMx must be 0 so the Timer_D frequency is ≤ 256 MHz. The illegal condition *TDHMx = 0:TDxCLn ≤ 0x0007* sets the minimum value of $TDxCLn_{MIN}$:

$$TDxCLn_{MIN} = 8 \tag{23}$$

and,

$$TDxCL0_{effective} = 2 * TDxCLn_{MIN} - 1 = 15 \tag{24}$$

so, for Up mode,

$$PWM_{frequency_{MAX}} = \frac{\text{Frequency}_{\text{TIMERD}}}{TDxCL0e_{effective_{MIN}} + 1} = \frac{25\ MHz * 8}{16} = 12.5\ \text{MHz}\ (\text{for TDHM} = 0) \tag{25}$$

Note from the last two examples that with a multiplier of 8, a PWM frequency higher than that achieved with a multiplier of 16 can be generated. This fact may sound counter-intuitive at first and it is due to the characteristics of TDxCL0 for each of the two multipliers.

Finally, the same example in free-running mode is worth investigating:

Hi-res Mode = Free running

In order to obtain the maximum PWM frequency, it would be ideal to achieve a Timer_D frequency of 256 MHz with TDHMx = 0. As shown in the previous example, when TDHMx = 0, $TDxCLn_{MIN} = 8$. The Timer_D Local Clock Generator Frequency table in the device-specific data sheet shows how to achieve different frequency ranges for Timer_D. As an example, the data sheet for MSP430F5172 is used. The following combination achieves a minimum frequency of 212 MHz in all devices with TDHMx = 0:

TDHREGEN = 0, TDHMx = 0, TDHCLKCR = 1, TDHCLKRx = 2, DHCLKSRx = 31, TDHCLKTRIM = 64	212	303	394	MHZ

Note that the system must be calibrated in software so the Timer_D frequency does not exceeds 256 MHz. TLV constants may also be used for this example. With this configuration, the following is true:

$$TDxCLn_{MIN} = 8\ since\ TDHMx = 0$$

$$TDxCL0effective_{MIN} = 2 * TDxCLn_{MIN} - 1 = 15\ \ (for\ 50\%\ duty\ cycle)$$

$$PWM_{frequency_{MAX}} = \frac{\text{Frequency}_{\text{TIMERD}}}{TDxCL0_{effective_{MIN}} + 1} = \frac{256\ MHz}{15 + 1} = 16\ \text{MHz} \tag{26}$$

Hence, the absolute maximum PWM frequency is 16 MHz for 50% duty cycle. This is also true in regulated mode with an input clock source clock of 32 MHz. Hence, an external crystal is required in regulated mode to obtain a 16 MHz PWM signal, while no crystal is needed in free-running mode. Note that in normal operation (non hi-res), the maximum frequency is 25 MHz/2 = 12.5 MHz; however the period and duty cycle resolution in normal mode is drastically reduced.

5.3 PWM Duty Cycle Resolution

The important characteristic of TDxCL0 regarding the four LSB of the TDxCL0 in the 16x case (TDHM = 1) or the three LSB in the 8x case (TDHM = 0) is not a characteristic of TDxCLn registers. Thus, the resolution of the duty cycle for a fixed PWM frequency is given by:

$$PWN_{DutyCycle_{resolution}} = 1/\text{Frequency}_{\text{TIMERD}} \tag{27}$$

The maximum PWM duty cycle resolution for both, regulated and free-running mode, is obtained for $Frequency_{TIMERD_{MAX}} = 256\ MHz$:

$$PWN_{DutyCycle_{resolution}} = \frac{1}{\text{Frequency}_{\text{TIMERDMAX}}} = 4\ \text{ns} \tag{28}$$

The limitations on duty cycle for a PWM signal were presented above and given by:

$$TDHMx = 0 : TDxCLn \le 0x0007$$
$$TDHMx = 0 : \ TDxCLn \le 0x0007$$
$$TDxCLn \ge TDxCL0 - 0x0008 \quad (29)$$

5.4 Generating Variable Out-of-Phase PWM Signals

Multiple TDxCLn registers can be used to generate multiple variable-frequency-fixed-duty-cycle PWM signals with variable relative phases. The number of TDxCLn registers required to generate *M* PWMs with variable relative phases is *M+1*, since TDxCL0 is always used to set the period of all PWM outputs.

Variable-phase-shift PMW pair generation is also achievable with Timer_A and Timer_B; however, the phase shift resolution of Timer_D is much higher. The generation of two 50%-duty cycle PWM signals with a relative phase shift from 0° to 180° is possible with a maximum resolution of 4ns, since $PWM_{DutyCycle_{resolution}} = 4\ ns$.

The period of the PWM output is determined by the value of TDxCL0. For two PWM signals with relative phase shift to be generated TDxCL1 and TDxCL2 can be used. The relative shift is achieved by maintaining one of the TDxCLn values constant and varying the other TDxCLn value. Figure 13 shows an example that keeps TDxCL1 constant and varies TDxCL2 to generate a relative phase shift between TDx.1 and TDx.2. The OUTMODx of TDx.1 and TDx.2 must be set to toggle mode; otherwise, changing TDxCLn results in a duty cycle change instead of a relative phase shift.

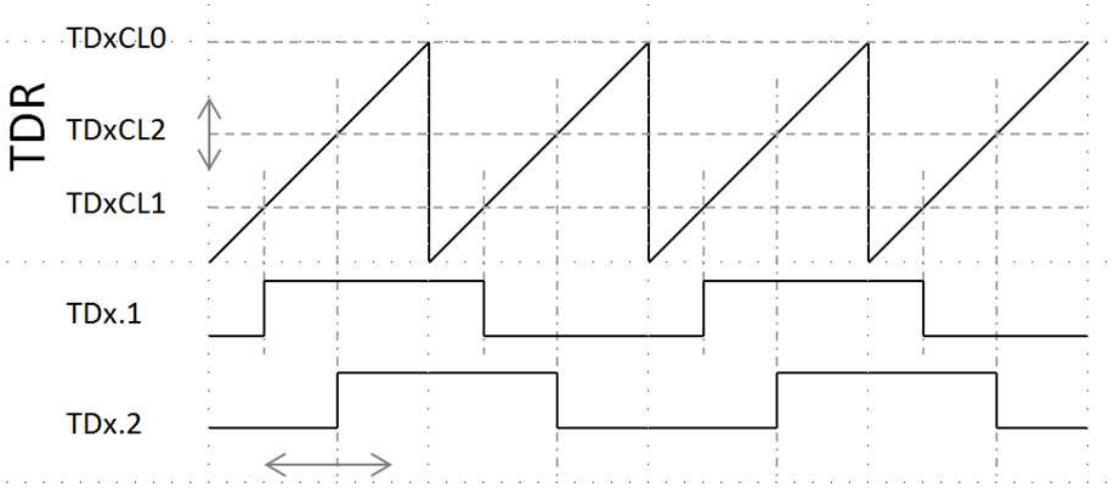

Figure 13. Generating Two PWM Signals With Relative Phases

A limitation exists when using Timer_D to generate more than one PWM output with relative phase shifts. The described application works well for PWM frequencies in the kHz range. As the output frequency increases (by decreasing TDxCL0), a top-level routing issue becomes apparent. It is shown in the device-specific data sheet that there are pins that are preferred and others that are more relaxed. This fact results in a small shift between two outputs that theoretically are configured to be in phase (0° phase shift).

To illustrate this behavior, consider the following example:

Timer Mode: Free running,Up
Output Mode: Toggle on TD0.0 and TD0.1
CCR0 value = 47
CCR1 value = 47

This configuration should yield two PWM signals, on TD0.0 and TD0.1, that are is phase. However, a small phase shift is observed as shown in Figure 14.

Figure 14. Generated Waveforms TD0.0 in blue and TD0.1 in Green

6 Choosing an Operation Mode

One of the main challenges when using Timer_D is to determine which operation mode fits the applications needs best. Three options are available:

- Normal Mode (non hi-res)
- High-Resolution Mode – Free Running
- High-Resolution Mode – Regulated

Selecting an operation mode depends on several factors including timer clock resolution and accuracy requirements, hardware and software resources available and overall final application. This section is meant to aid designers in selecting the most efficient operation mode when working with Timer_D. It is impossible to present all scenarios; hence, a general analysis is introduced.

Suppose a PWM signal is required. Usually the designer knows the following requirements:

- PWM frequency and duty cycle
- Minimum tolerance
- Minimum resolution
- Hardware resources available

The first mode to consider should be the basic (non hi-res) since it is the easiest one to configure, it is the most power efficient in active mode and it requires the least memory resources for configuration. If the basic resolution $\left(\frac{1}{source\ clock}\right)$ and maximum PWM frequency $\left(\frac{source\ clock}{2}\right)$ are good enough for the final application, then evading the high-resolution mode may be the most efficient way to achieve the PWM signal.

When a higher resolution is required, two options are available. Regulated mode is easier to configure in software since TDHCLKRx, TDHCLKSRx, and RDHCLKTRIMx are hardware controlled and no software-based calibration is needed, but the target frequency is harder to achieve since the period resolution is controlled by TDxCL0. Regulated mode may be a good option for applications with relaxed tolerances. Free-running mode is the most complex from a software perspective, but provides the best resolution and the most flexibility. Free-running also enables the timer (and, hence, the PWM) to be ON in any low power mode. For applications that lack an external crystal and require a minimum tolerance of ±1.5%, free-running mode may be the best option. Table 8 highlights the advantages and disadvantages of all operation modes.

Table 8. Advantages and Disadvantages of Timer_D Operation Modes

Operation Mode	Advantages	Disadvantages
Basic	Easiest configuration	Lowest resolution (1/source clock)
	No calibration required – tolerance set by source clock	External crystal required to run timer in LPMx
	Easy to migrate to Timer_A and Timer_B	Maximum PWM frequency = source clock/2
Hi-Res Free-Running	No clock source required external to Timer_D module	Timer_D local clock is highly dependent on voltage, temperature and varies across devices
	Maximum frequency is independent of system frequency	
	Better PWM period resolution through TDHCLKRx, TDHCLKSRx,and RDHCLKTRIMx	Frequency calibration may be required in software
	Use of TLV constants	Complex software configuration
	Runs in LPMx without external crystal	
	Higher maximum PWM frequency (16 MHz)	
Hi-Res Regulated	Easy software configuration	Clock source required external to Timer_D module
		PWM period resolution limited by TDxCL0 resolution
	No software calibration required – tolerance set by source clock	32 MHZ external crystal needed to achieve 16 MHz PWM signal
		External crystal required to run timer in LPMx

7 TIMER_D High-Resolution – Summary

Table 9. Free-running vs. Regulated Mode

	Free-Running Mode	Regulated Mode
Clock Source	Internal to Timer_D module	External to Timer_D module
Frequency Control	High granularity with software control of TDHMx, TDHCLKRx, TDHCLKSRx, and RDHCLKTRIMx	Input clock times multiplier (8 or 16). TDHMx control only
Calibration required?	Yes	No

8 FAQ

This section covers the most common questions regarding the high-resolution mode of Timer_D.

Q1) In Regulated mode, how can I set TDHCLKRx, TDHCLKSRx, and TDHCLKTRIMx bits in software?

A1) The hi-resolution generator of the Timer_D can be used in either free-running or regulated mode. The TDCALEN bit enables selection between these two modes. With TDCALEN = 1, regulated mode of the hi-resolution clock generator is used and the TDHCLKRx, TDHCLKSRx, and TDHCLKTRIMx bits are controlled by hardware to automatically reach the target frequency and cannot be modified in software.

Q2) What is the maximum frequency of Timer_D?

A2) The maximum frequency of Timer_D is 256 MHz. In regulated mode, for an input clock above 16 MHz the maximum multiplier is x8 to ensure the frequency limitation is not exceeded. In free-running mode, software must ensure the configuration of the timer maintains the frequency with an upper limit of 256 MHz. Due to the variations and the desire to meet at least 256 MHz under all circumstances the design must cover a wide range of frequency trimming. Consequently, some parameters lead to >1 GHz in some devices as shown in Table 10. Any frequency configuration that yields a Timer_D frequency above 256 MHz is provided for documentation purposes only and the programmer must ensure the maximum frequency of 256 MHz specification is met to ensure all parts of the system to operate properly.

Table 10. Invalid Timer_D Configuations (over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted)

Parameter		Test Conditions	Min	Typ	Max	Unit
$f_{HRCG(2,31,128)}$	HRCG frequency (2, 31, 128)	TDHREGEN = 0, TDHMx = 0, TDHCLKCR = 1, TDHCLKRx = 2, TDHCLKSRx = 31, TDHCLKTRIM = 128	290	413	537	MHz
		TDHREGEN = 0, TDHMx = 1, TDHCLKCR = 1, TDHCLKRx = 2, TDHCLKSRx = 31, TDHCLKTRIM = 128	580	826	1074	MHz

Q3) If the value of TDCCRx is changed in software, when does the value get loaded into TDCLx?

A3) The compare latch, TDCLx, holds the data for the comparison with the timer counter value in compare mode. TDCLx is a capture and compared register that is accessible to the user software. The buffered compare latch gives you control over when a compare period updates. Compare data is written to each TDCCRx and automatically transferred to TDCLx. The timing of the transfer from TDCCRx to TDCLx is user selectable by setting the CLLDx bits:

Table 11. TDCLx Load Events

CLLDx	Description
00	New data is transferred from TDCCrx to TDCLx immediately when TDCCRx is written to.
01	New data is transferred from TDCCrx to TDCLx immediately when TDCCRx *counts* to 0.
10	New data is transferred from TDCCrx to TDCLx immediately when TDCCRx *counts* to 0 for up and continous modes. New data is transferred to or from TDCCRx to TDCLx when TDxR *counts* to the old TDCL0 or to 0 for up and down modes.
11	New data is transferred from TDCCrx to TDCLx when TDxR *counts* to the old TDCLx value.

Q4) What is the power consumption of Timer_D is Active and LPMx modes?

The power consumption for Timer_D in free-running mode at 3 V was measured and results were summarized in Table 12.

Table 12. Timer_D Power Consumption (1)

Condition	Active (µA)	LPM0 (µA)	LPM1 (µA)	LPM2 (µA)	LPM3 (µA)	LPM4 (µA)
Timer_D OFF	160	83	81	9	4	1
Timer_D ON, PWM OFF	733	656	654	610	608	605
Timer_D ON, PWM ON	1279	1033	1031	984	981	978

(1) Power numbers are independent of Timer_D frequency.

9 References

1. *Two-Dimensional Capacitive-Touch Implementation Using the High-Resolution Timer_D of the MSP430F5132* (SLAA418)
2. *PWM DAC Using MSP430 High-Resolution Timer* (SLAA497)
3. *Intelligent LED lighting control with the ultra-low-power MSP430F51x2 microcontrollers* (SLAY024)
4. *MSP430F51x1, MSP430F51x2 Mixed Signal Microcontroller Data Sheet* (SLAS619)
5. *Timer_D Module (Chapter Excerpt From MSP430x5xx and MSP430x6xx Family User's Guide, SLAU208)* (SLUA323)
6. *MSP430x5xx and MSP430x6xx Family User's Guide* (SLAU208)

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265