Application Report
SLAA654A–November 2014–Revised July 2015

Designing With MSP430™ MCUs and Segment LCDs

Katie Pier *MSP430 Applications*

ABSTRACT

Segment liquid crystal displays (LCDs) are needed to provide information to users in a wide variety of applications from smart meters to electronic shelf labels (ESL) to medical equipment. Several MSP430™ microcontroller families include built-in low-power LCD driver circuitry that allows the MSP430 MCU to directly control the segmented LCD glass. This application note helps explain how segmented LCDs work, the different features of the various LCD modules across the MSP430 MCU family, LCD hardware layout tips, guidance on writing efficient and easy-to-use LCD driver software, and an overview of the portfolio of MSP430 devices that include different LCD features to aid in device selection.

Contents

List of Figures

Trademarks

MSP430 is a trademark of Texas Instruments.
All other trademarks are the property of their respective owners.

1 Introduction: MSP430 + LCD End Applications

There are a number of common applications where MSP430 microcontrollers with built-in LCD drivers are a great fit. This can be any application where you need a segmented LCD display, but battery life or current consumption is important. Examples include low-power LCD handhelds (like a watch or other device), blood glucose meters, appliances, water meters, electronic shelf labels, and one-time password tokens. The combination of rich analog and peripheral interfaces provided by MSP430 devices, along with the built-in segment LCD display driver, enable a diverse array of applications with a compelling set of features all in one system on chip (SOC).

2 MSP430 LCD Portfolio

Table 1 compares the LCD modules that are available on MSP430 MCUs.

Table 1. MSP430 LCD Module Feature Comparison

Parameter	LCD	LCD_A	LCD_B	LCD_C	LCD_E
Number of segments supported[(1)]	128/4-mux	160/4-mux	160/4-mux	320/8-mux	448/8-mux
Mux mode supported	4, 3, 2, 1	4, 3, 2, 1	4, 3, 2, 1	8, 7, 6, 5, 4, 3, 2, 1	8, 7, 6, 5, 4, 3, 2, 1
Segment functionality against port pin selection	Minimum is group of 16	Groups of 4 segments	Groups of 4 segments	Individual selection	Individual selection
Flexible configuration for COM and Segment pins	NO	NO	NO	NO	YES
LCD clock selection	ACLK	ACLK	ACLK, VLO	ACLK, VLO	ACLK, XT1, VLO
LCD clock divider availability	NO	32 to 512 (8 settings with 32 counts apart)	1 to 1024 (192 settings with 111 unique dividers)	1 to 1024 (192 settings with 111 unique dividers)	8 to 2048 (depends on Mux mode)
Interrupt capabilities	NO	NO	YES (4 sources)	YES (4 sources)	YES (3 sources)
Whole display blinking	Manual only	Manual only	YES	YES	YES
Programmable blinking frequency	NO	NO	YES	YES	YES
Individual segment blinking capabilities with separate memory	NO	NO	YES	YES	YES
Dual memory display	NO	NO	YES	YES	YES
LCD bias generation using resistive network	External	External or Internal	External or Internal	External or Internal	External or Internal
Device protection against no connected capacitance on LCDCAP when charge pump is used	NO charge pump	NO (A 4.7-µF or larger capacitor must be connected from LCDCAP to GND)	Protected with LCDNOCAPIFG interrupt flag	Protected with LCDNOCAPIFG interrupt flag	NO need for protection (A 0.1-µF or larger capacitor must be connected from LCDCAP0 and LCDCAP1 pins)
Charge pump voltage with external voltage reference	NO charge pump	$3 \times V_{ref}$	Programmable (15 levels)	Programmable (15 levels)	Programmable (15 levels)
Low-power waveforms mode	NO	NO	YES	YES	YES

(1) LCD pin count varies with device and package. See device-specific data sheet for detailed information.

3 Segmented LCD Operation

The following sections explain the basic operation of all LCDs. This helps create a background for our later discussion of MSP430 LCD driver features.

3.1 LCD Structure (Simplified)

Figure 1 shows a simplified version of the structure of a segment LCD display. Essentially it consists of two polarizers rotated 90 degrees from each other to polarize light coming into the display, liquid crystals between the polarizers with electrodes to apply a charge, and a reflective backing to reflect light that gets through all the layers of the display.

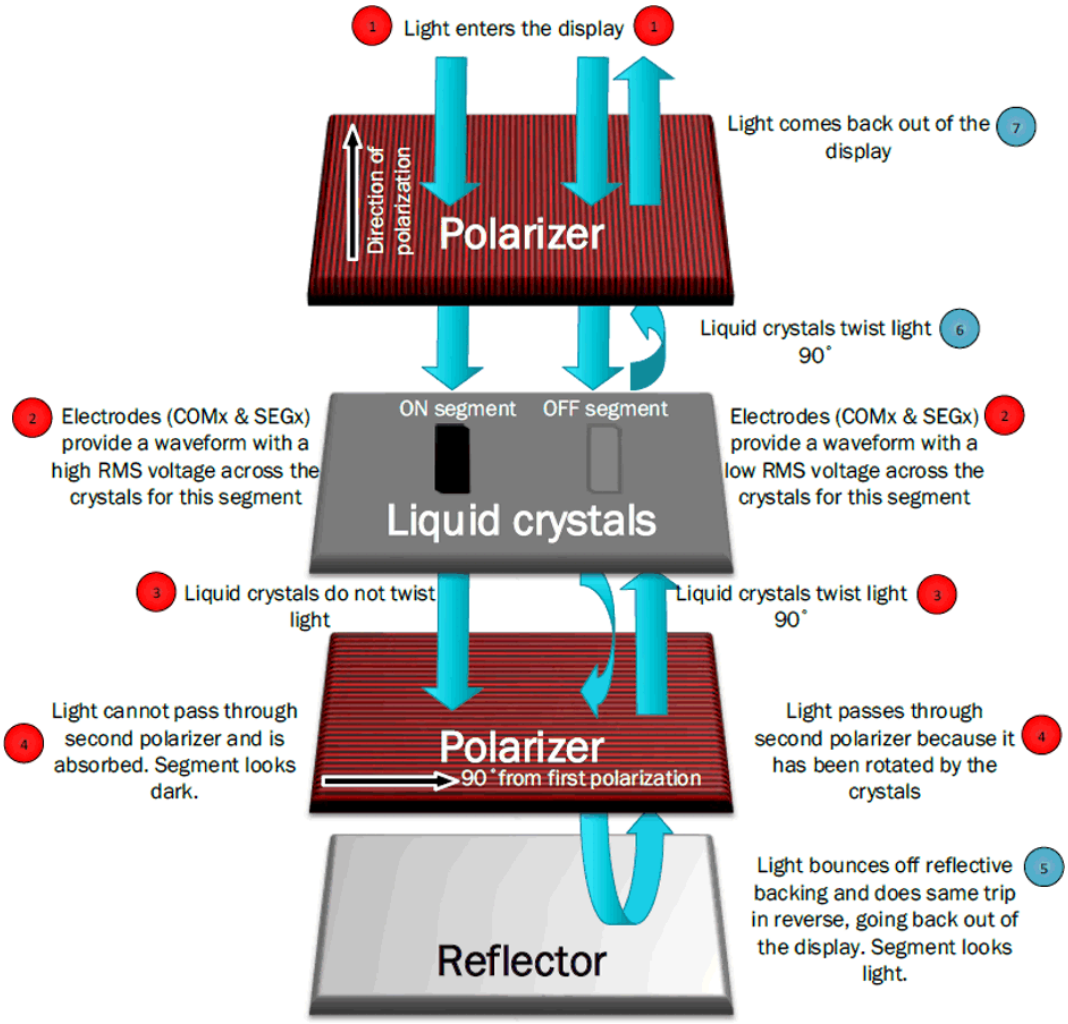


Figure 1. Segmented LCD Structure and Theory

When no charge is applied to the electrodes for a particular segment, the segment is "off" or gray. In this normal state, the liquid crystals have a twisted structure that turns the light 90 degrees. So when no charge is applied: first, light comes in the first polarizer and comes out polarized in one direction. Then, the crystals turn the light 90 degrees as it passes through them – this allows the light to be able to pass through the second polarizer because it is rotated compared to the first one. Finally the light reflects off the reflective backing and does the same path in reverse. Because the light is reflected back, it looks light or gray.

When a charge is applied to the electrodes for a segment, the segment is "on" or black. In this charged state the crystals untwist, so they do not turn the light as it passes through them. So when a charge is applied: first, light comes in the first polarizer and comes out polarized in one direction. Then, the crystals simply allow the light to pass straight through without turning it this time. Because the second polarizer is at 90 degrees from the first one, the light is not able to pass through and is instead absorbed. This makes the segment look dark.

3.2 LCD Drive Basics

LCDs must be driven with AC signals. A DC level on an LCD segment will damage the LCD – typically less than 50-mV DC voltage is allowed. The MSP430 LCD module generates these types of AC waveforms automatically so that the user only has to specify if a segment should be on or off – the internal hardware does the rest.

LCD segments have a charge applied to the crystals between two electrodes – a COMx line and an Sx segment line. The potential difference applied by these two electrodes is the waveform seen by the LCD segment.

The RMS voltage presented on an LCD segment determines whether it is on or off. The example waveforms in Figure 2 show resulting waveforms (combination of COMx and Sx pin signals) of both an on and an off segment. The on segment has a much larger RMS voltage applied on the segment than the off segment. Note that both segments have waveforms that have net zero DC voltage, but the RMS voltage of the on segment is higher, which causes the segment to turn on and look dark.

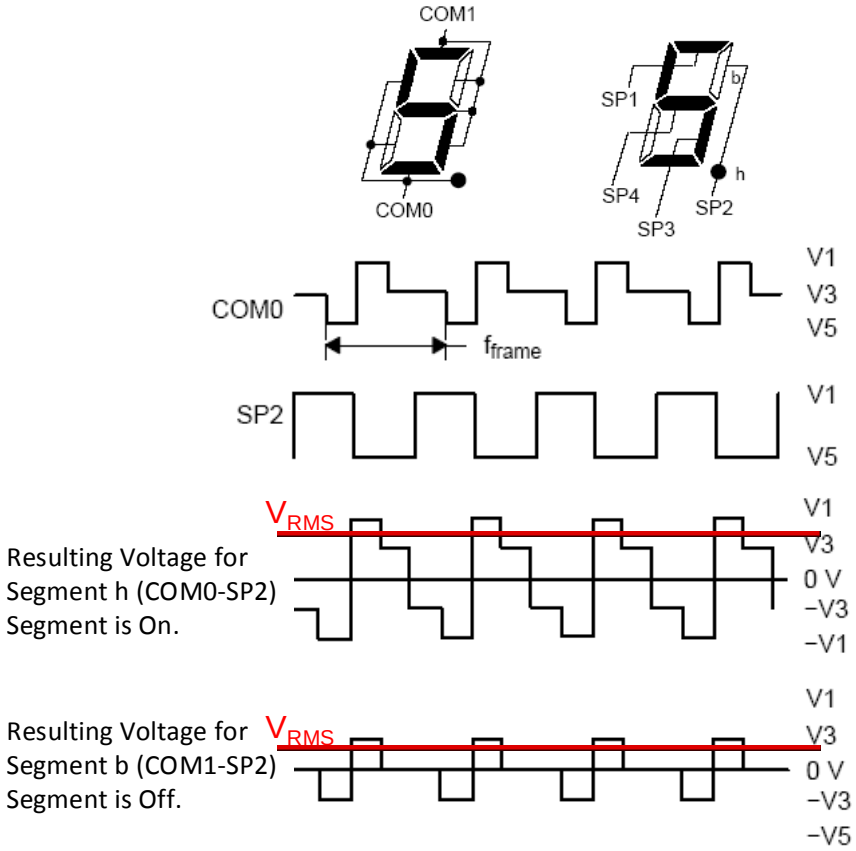


Figure 2. Example LCD AC Waveforms

4 MSP430 LCD Features

This section explains the different features available on MSP430 LCD modules. Different MSP430 LCD modules have different combinations of features, so always make sure to check the data sheet for the particular device you are using to see what LCD module is present in the device and how many pins are available for LCD output. Chapter 2 goes through the MSP430 LCD module portfolio to help with device selection.

4.1 Muxing

Segmented LCDs use multiplexing (muxing) to limit control pin count. Types of displays include static (no muxing), 2-mux, 3-mux, 4-mux, etc. – even up to 8-mux. The notation N-mux means each segment pin Sx drives N segments on the display – this also means there are N common (COMx) pins. Each LCD segment on the display is driven by the combination of a COMx pin and Sx pin, providing a difference in potential across the liquid crystals for that segment.

Muxing allows a much larger number of segments to be controlled by a limited number of pins. If for example there is an 8-mux LCD display, then there are 8 COM pins and each segment (Sx) pin can drive 8 segments. So for example when using an 8-mux capable MSP430 with 40 Sx pins available (S0-S39), then it could control 320 segments with only 8 (COMx) + 40 (Sx) = 48 pins.

Some MSP430 devices with 8-mux mode can support up to 320 segment displays. However, always make sure to check the device-specific data sheet to see how many segments the particular device supports, as it is limited not only by the muxing capability of the LCD module but also by the number of LCD pins available on the particular device in a particular pin-package.

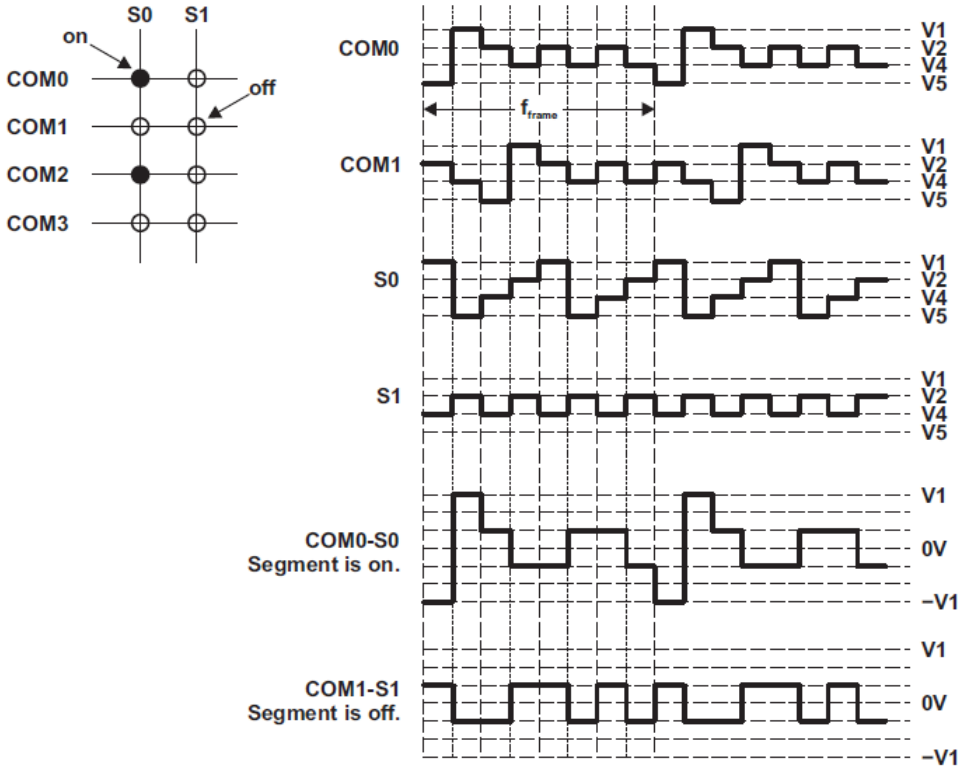


Figure 3. 4-Mux Connections and Waveforms

4.1.1 Muxing Example

Figure 4 shows a basic 2-mux example. Each segment is controlled by signals on 2 pins in this case – a COMx pin and an Sx segment pin. The signals shown are the waveforms applied on the electrodes for the segment, so the potential difference between the Sx and COMx signal is what is being applied to the liquid crystals in that area. This potential difference is what is shown as the voltage in the resultant waveforms in Figure 4 (COM0-S0, and COM1-S1).

In this example, the COM0-S0 waveform has a high RMS voltage so the segment is on even though the waveform has a net zero DC voltage. The COM1-S1 waveform has a low RMS voltage, so it is off. While the waveforms may look complex, remember that they are generated automatically by the MSP430 LCD module – the user just has to specify the basic settings of the LCD, and then indicate which segments should be on or off.

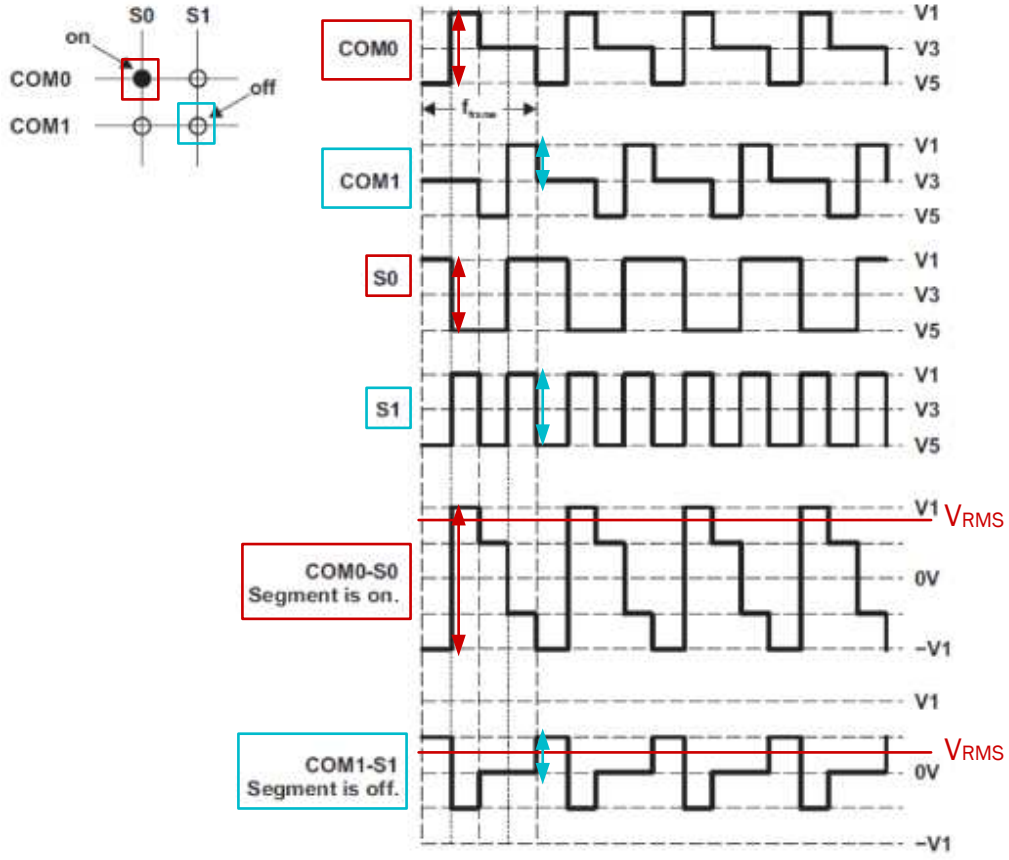


Figure 4. 2-Mux Example

4.2 Charge Pump

The V_{LCD} voltage sets the voltage level of V1, the highest LCD voltage in the waveforms. This can be set in software to be sourced from AVCC, the internal charge pump, or from an external source. Most MSP430 LCD modules (LCD_A, LCD_B, LCD_C, and LCD_E) include a built-in charge pump – only the original LCD module on some F4xx devices does not have one.

The advantage of using a charge pump for generating the V_{LCD} voltage is that: 1) it provides a regulated voltage to the LCD to keep a stable voltage output for the display, and 2) it allows you to have the V_{LCD} set to a different voltage level that is independent of D_{VCC} – this lets you set V_{LCD} to the best level for the particular LCD display, and keep good contrast even as the battery in the system drains.

The built-in charge pump has programmable voltage levels for use with different segmented displays – the allowed maximum operating voltage is something that would come from the design of the particular LCD glass that is being used. Setting a different V_{LCD} can change your contrast ratio, so having software-configurable voltage levels from the charge pump allows contrast control via software. For example, using a lower V_{LCD} may provide less contrast, but also have less current consumption, so this is a trade-off to experiment with in a final design.

The charge pump requires an external capacitor for operation. If this capacitor is not present and the charge pump is turned on, the MSP430 device could be damaged. Because of this, some of the newer LCD modules like LCD_B and LCD_C include detection circuitry that automatically disable the charge pump if no capacitor is present, as well as setting an interrupt flag to warn user software of this condition. Note that the LCD_E module has a different configuration for the charge pump capacitor (connected between two pins instead of directly to ground), so it is not damaged if the capacitor is not present (though the charge pump does not work properly).

The charge pump can also be referenced to follow an external source. This is useful if multiple MSP430 MCUs are used together to control a single larger segmented display than could be controlled with a single device. The application note *Use of Two MSP430 Microcontrollers to Enhance Segment Lines for Larger LCDs* (SLAA072) describes how to do this.

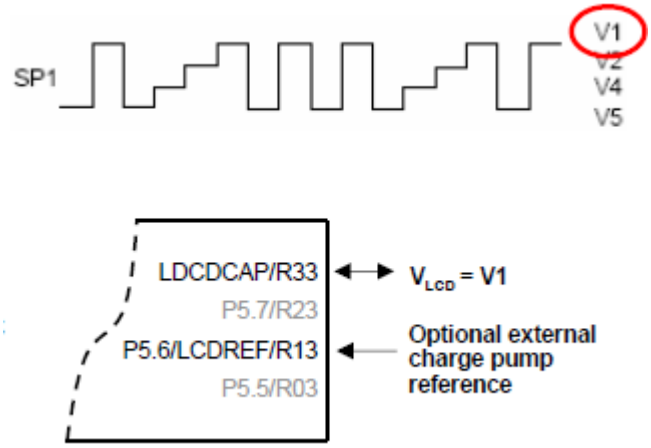


Figure 5. Charge Pump

4.3 Biasing

The highest voltage level V1 is generated by V_{LCD}. V_{LCD} can be sourced either externally, or from the internal charge pump as discussed in Section 4.2. To produce the rest of the voltage levels in the LCD waveforms, V2 through V5, the module needs to be able to produce bias voltages at fractions of V_{LCD}.

The bias voltages V2 through V5 can be divided down from V_{LCD} either internally or with an external resistor network. This selection is entirely independent of how V_{LCD} is being sourced (there can be any combination of internal/external V_{LCD} source, and internal/external bias voltage generation). Depending on the muxing being used and the specific MSP430 device, different bias options like 1/2 or 1/3 may be available – check the device specific data sheet and user's guide. For 1/2 bias mode, voltage levels V1, V3, and V5 are used in the waveforms – V1 = V_{LCD}, V3 = 1/2 V_{LCD}, and V5 = 0. For 1/3 bias mode, voltage levels V1, V2, V4, and V5 are used in the waveforms – V1 = V_{LCD}, V2 = 2/3 V_{LCD}, V4 = 1/3 V_{LCD}, and V5 = 0. Figure 6 shows an example of the possible internal and external bias options in one of the LCD modules.

Generating bias voltages internally is simple because no external components are required – the module internally divides down the voltage. However, generating bias voltages externally instead can be lower power. External biasing requires the user to provide an external resistor divider to create the voltages V2 through V5 - the resistor divider used depends on the biasing mode – static, 1/2, or 1/3 bias as you can see in Figure 6. The resistors in the divider should all be the same value, but the size used may depend on the particular display used in the design.

Changing the external resistor values can impact both current consumption and contrast – see the application note *Driving Large LCD With LCD Peripheral of the MSP430* (SLAA272) for more details on these tradeoffs and selecting the size of the resistors for the divider. Larger resistors cause less current consumption in the resistor ladder, saving you power – however, if resistors are too large, the contrast may not be good or may not be even for all segments. Experimentation with different sizes of resistor is usually needed in a design to find the best combination of performance vs current consumption.

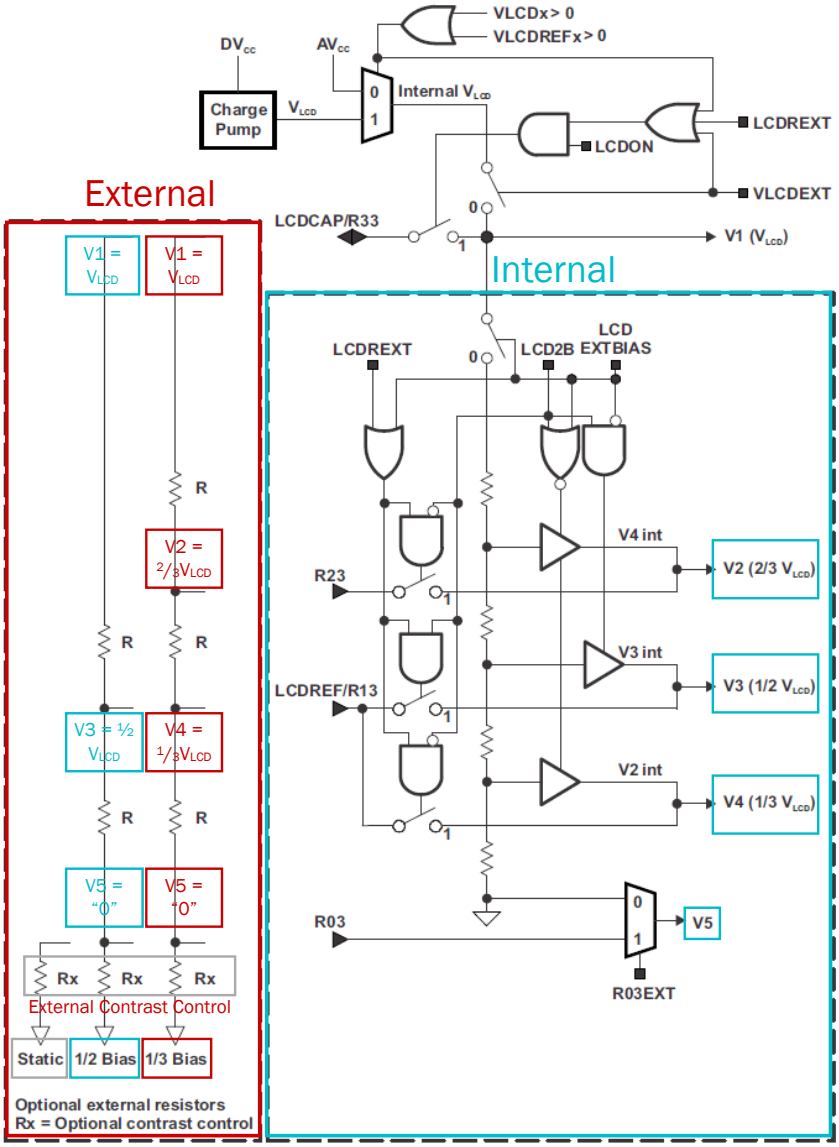


Figure 6. Bias Configurations

4.4 Contrast Control

As mentioned in Section 4.2, when using the charge pump the V_{LCD} of the display is software controlled. This allows the user to easily adjust contrast in software. Changing V_{LCD} adjusts all of the other LCD voltages as well regardless of internal or external biasing, because all of the other voltages are simply divided down from V_{LCD}.

When using R03EXT bit to source V5 voltage externally, contrast can also be adjusted in hardware by changing the optional resistor Rx on the R03 pin in Figure 6 – this changes the voltage at the low end of the resistor divider. When using an external bias resistor ladder for generating the bias voltages, the sizing of R can also have an impact on contrast. If contrast is not even across all segments, the size of the resistors in the bias ladder may need to be reduced at the tradeoff of additional current consumption (see Section 4.3 on biasing).

The different biasing modes and the particular LCD being used can also have an impact on contrast ratio. As you can see in Table 2, the contrast ratio can be represented as $V_{RMS,ON}/V_{RMS,OFF}$, or the RMS voltage from the waveforms for a segment that is on, divided by the RMS voltage from the waveforms for a segment that is off. The higher the contrast ratio, the greater the difference in appearance of an on segment versus an off segment. Table 2 shows that there is better or worse contrast depending on the bias configuration and muxing of the display – this is because these settings affect the characteristics of the waveforms that are output. As shown in the table, higher mux rates tend to have more lower contrast ratios, which means that their performance is more sensitive to any tradeoffs that affect contrast – this means that a more sensitive LCD glass with a better threshold, or other factors to provide better contrast (such as higher V_{LCD}, smaller bias resistors, or the techniques from Section 4.10), might be needed for the desired LCD performance.

A typical approach to determine the V_{LCD} for good contrast, is to use the threshold voltage when the contrast is 10% and use this with the $V_{RMS,OFF}/V_{LCD}$ ratio from the user's guide table to calculate a recommended V_{LCD} using this equation: $V_{LCD} = V_{th,10\%}/(V_{RMS,OFF}/V_{LCD})$. The $V_{th,10\%}$ is a characteristic of the fluid used in the LCD display, so it varies with the display. The display information provided by the manufacturer typically lists a visual threshold voltage for 10%.

Some configurations trade off a reduced contrast ratio for a reduction of the full-scale LCD voltage V_{LCD} used. For example, on some modules 1/3 bias may give a better contrast, but the 1/3 bias mode may require a higher V_{LCD} to be used as well. See the LCD module-specific section in the family user's guide for more information pertaining to the particular LCD module and contrast ratio – the user's guides have tables like the one below with information specific to that module's muxing and bias options (Table 2 shows information for LCD_B).

Table 2. LCD Voltage and Biasing Effect on Contrast

Mode	Bias Configuration	Voltage Levels	$V_{RMS,OFF}/V_{LCD}$	$V_{RMS,ON}/V_{LCD}$	Contrast Ratio $V_{RMS,ON}/V_{RMS,OFF}$
Static	Static	V1, V5	0	1	1/0
2-mux	1/2	V1, V3, V5	0.354	0.791	2.236
2-mux	1/3	V1, V2, V4, V5	0.333	0.745	2.236
3-mux	1/2	V1, V3, V5	0.408	0.707	1.732
3-mux	1/3	V1, V2, V4, V5	0.333	0.638	1.915
4-mux	1/2	V1, V3, V5	0.433	0.661	1.528
4-mux	1/3	V1, V2, V4, V5	0.333	0.577	1.732

4.5 Timing

Most MSP430 LCD modules (LCD_A, LCD_B, LCD_C, and LCD_E) include internal timing generation within the module without having to use up any timer modules – only the original LCD module on some F4xx devices does not and instead requires the timing to be generated instead using the Basic Timer module.

The LCD module is sourced by a selectable clock that can then be further scaled and divided within the module to achieve the desired frequency for f_{LCD}. f_{LCD} frequencies are usually fairly low frequencies (<1 kHz), so typically the timing for the module is sourced from low-frequency ACLK, XT1, or VLOCLK – these clocks are also typically available even in some of the lowest low-power modes (LPM3 and LPM3.5). f_{LCD} is the frequency that generates the timing for the common COMx and segment Sx signals. The required f_{LCD} can be calculated using Equation 1.

$$f_{LCD} = 2 \times MUX \times f_{FRAME} \quad (1)$$

f_{FRAME} is the frame frequency from the data sheet for the LCD. The display typically has a range of allowed frame frequencies, which gives the user options when choosing an f_{LCD} to use. Lower frequencies give a lower current consumption, but higher frequencies give less flickering on the display. The user must weigh the tradeoff between performance and current consumption, and experimentation with different f_{LCD} frequencies can help determine what setting yields an acceptable appearance on the LCD with the least current consumption.

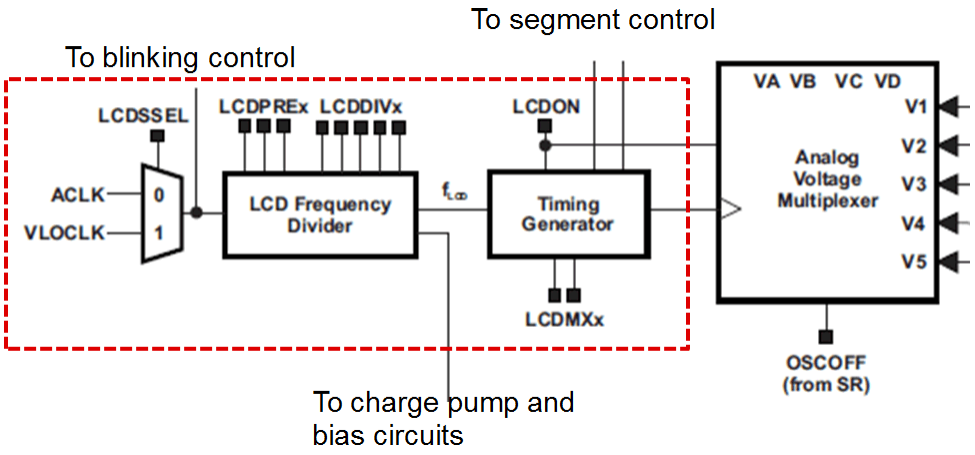


Figure 7. Timing Generation

4.6 Memory Map

User software selects which segments should be on or off by using the LCD memory registers. Each bit represents a single LCD segment connected to a COMx and Sx pin pair. The row (or byte) corresponds to the Sx pin, and the columns (or each bit within the byte) correspond to the COMx pins. In 2-mux through 4-mux modes, the upper and lower nibbles of each row correspond to different Sx pins – only up to 4 bits in the byte are needed since there are only up to 4 COMx lines, so two Sx pins can be set by a single byte. In 5-mux through 8-mux modes, there are more than 4 COMx lines so the whole row (byte) is required for each segment Sx pin.

Figure 8 shows an example memory configuration for 4-mux mode. The S38 and S39 segment pins in this example correspond to the lower and upper nibbles of the byte at LCD memory offset 0xA4. To control the segment connected to COM0 + S38 or COM0 + S39, the software would set the highlighted bits to either 1 or 0 to indicate the desired LCD segment state "on" or "off".

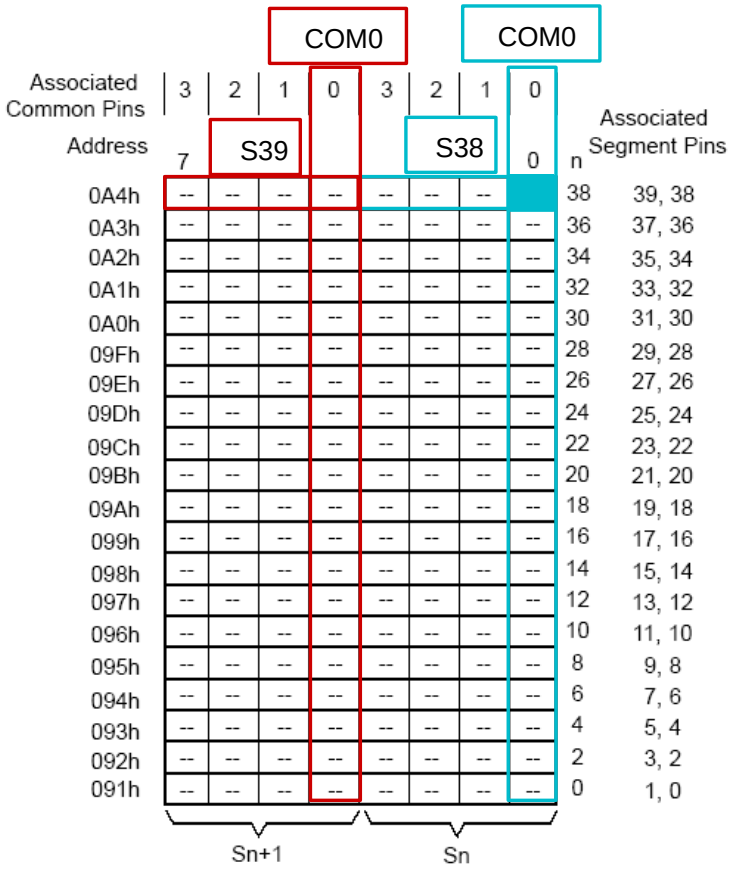


Figure 8. LCD Memory Map Example

4.7 Blinking

Most MSP430 modules support some form of blinking. This can be either blinking of the entire screen (LCD, LCD_A) or blinking of individual segments (LCD_B, LCD_C, LCD_E). For the original LCD and LCD_A module on F4xx devices, full-screen blinking can be handled manually by setting or clearing the LCDSON bit in software.

For devices that support both full screen and individual segment blinking (LCD_B, LCD_C, LCD_E) the blinking can happen automatically at a particular frequency. The blink frequency is configurable, but it must be less than the frame frequency.

When using individual segment blinking, whether a segment blinks or not is controlled by the blink memory. The structure of blink memory is just like LCD memory discussed in Section 4.6 and shown in Figure 8. The blink memory can also be used as a secondary display memory – the LCDDISP bit controls which memory is currently being used for the display. One of the blinking modes can also be used to automatically toggle between displaying the two memories.

For modules that support individual segment blinking, this feature may only be available in certain muxing modes, with whole screen blinking only being available in other modes – check the data sheet and user's guide for the specific MSP430 device for details.

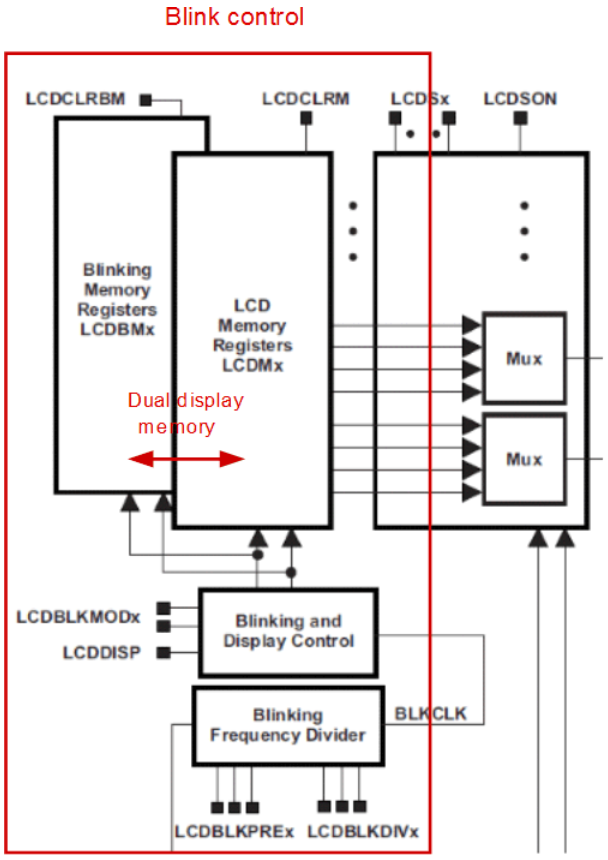


Figure 9. LCD Blinking and Dual Display Memory

4.8 LCD Output Pins

Some MSP430 microcontrollers have dedicated LCD pins – it is the only function available on these pins. However, some devices have LCD pins muxed with digital I/O functions. When the application is not using all of the LCD pins on these devices, these pins can be configured in software to be used for other functions. On devices with LCD_A, the pins can be configured for either GPIO or LCD function in groups of 4 pins, using a setting in the LCD module. On devices with LCD_B, LCD_C, and LCD_E, pins can be individually configured for either GPIO or LCD function using a setting in the LCD_B/C/E module registers.

On devices with the LCD_E module, the LCD pins are even more flexible. On these devices (FR4xx family), each pin can not only be configured for GPIO or LCD function, but each pin set for LCD function can further be configured as either a COMx or Sx pin. (On the other LCD modules, specific pins are set aside to be the COMx pins and the other pins are Sx pins). This extra configurability with LCD_E allows for completely configurable LCD pins, meaning users have the most flexible layout possible. This can help make sure the LCD layout can be achieved in a single layer, since the COMx pins on the MSP430 side are no longer fixed to a specific set of pins – they can be configured to help accommodate where the COMx pins are located on the display side. Further, some layout mistakes can be fixed in software instead of having to create a new PCB design. For more information regarding LCD layout, see Section 5.1. For more details about LCD_E flexible pin configuration, see the *MSP430FR4xx and MSP430FR2xx Family User's Guide* (SLAU445).

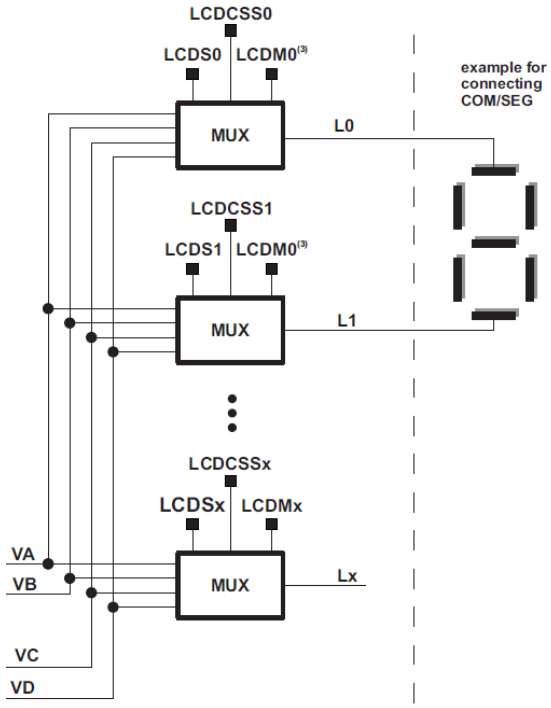


Figure 10. LCD_E Flexible COM and SEG Configuration

4.9 Ultra-Low-Power Features

MSP430 LCD modules are designed with ultra-low power as a key feature. In addition to some of the low power options discussed in the earlier sections (such as the adjustable charge pump voltage level, and options for external biasing), the charge pump is also only turned on for a small percentage of the overall operation of the module. It runs with a low duty cycle so that its peak current is only seen for a very small portion of the overall time that the LCD is on, helping the LCD to keep a very low overall average current.

The peak charge pump current can be found in the data sheet for the particular device being used. There is also usually a spec for the time to charge the C_{LCD} charge pump cap when it is discharged, so this can help determine an average current for the LCD module – worst case the charge pump is at the peak current $I_{CC,Peak,CP}$ for the time $t_{LCD,CP,on}$ when C_{LCD} has been discharged. The rest of the time, the module is in a much lower current state. In addition, using a low-leakage capacitor for C_{LCD} can also help to reduce the energy consumption as it helps the charge pump run with a lower duty cycle.

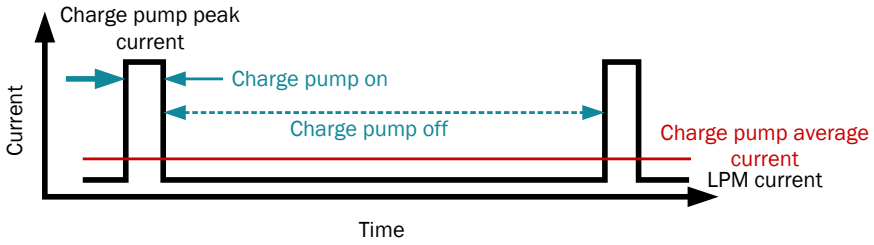


Figure 11. Low Charge Pump Duty Cycle

MSP430 devices with LCD_C or LCD_E modules also have a setting for using lower power versions of the LCD waveforms. The lower power versions of the waveforms have the voltage sequence re-shuffled such that certain timeslots are grouped together. This makes for fewer switching events on each pin and lower current consumption. Figure 12 shows an example of the normal and lower power versions of waveforms for 8-mux mode.

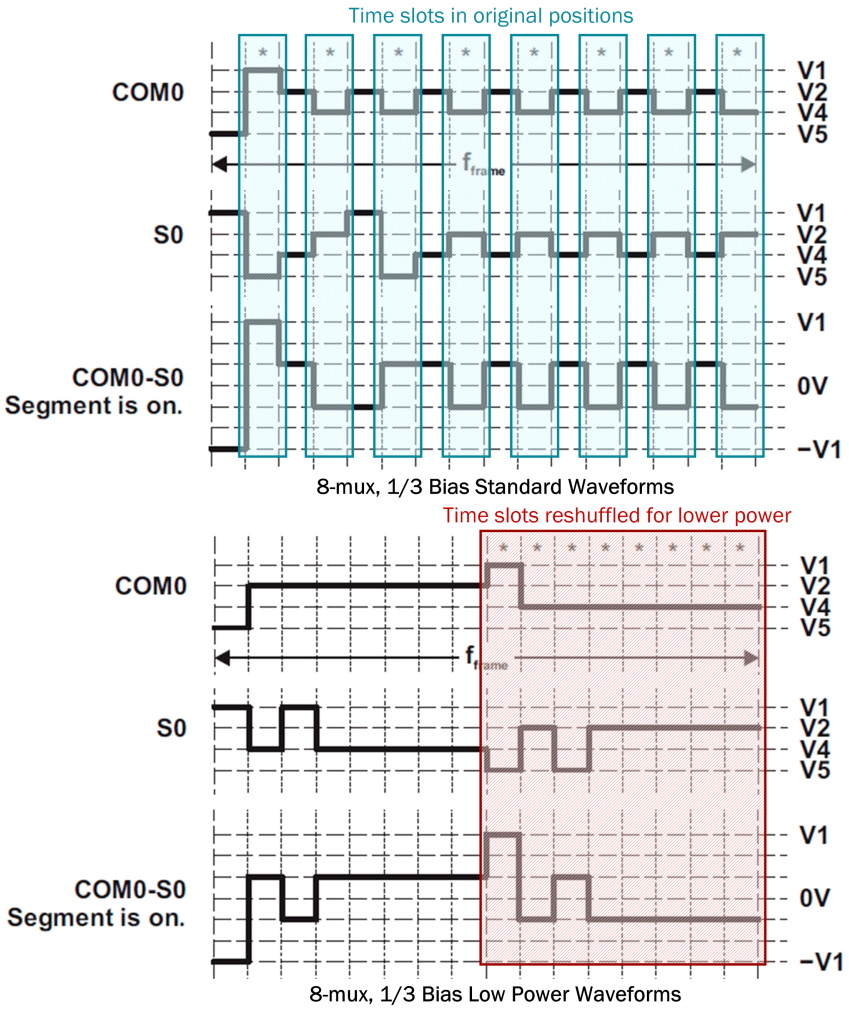


Figure 12. Low-Power Waveforms Example

The LCD_E module is currently the lowest power LCD module in the MSP430 portfolio and is designed to be even lower power than the other MSP430 LCD devices. One key feature is that the module can remain operational and keep the LCD on all the way down to LPM3.5 mode, allowing for a new level of ultra-low power and enabling new LCD applications where the power supply is very limited. See the device-specific data sheet and user's guide for more details about LCD current consumption and LPM3.5 mode.

4.10 Driving Large LCDs

Large LCD panels consume more current – each segment is like a capacitor that is constantly being charged and discharged, so larger displays (with larger segments) have a greater capacitance which means more charge for each cycle of the display. It is also harder to keep good contrast while driving a larger LCD glass. There are some hardware techniques that can be used to help improve LCD performance and contrast for large digit size LCD displays using MSP430 LCD modules. These include adding caps to the bias resistor network to reduce ripple, or reducing the resistor ladder values (with a tradeoff of increased current). There is a detailed discussion of these techniques in the application note *Driving Large LCDs with LCD Peripheral of the MSP430* (SLAA272).

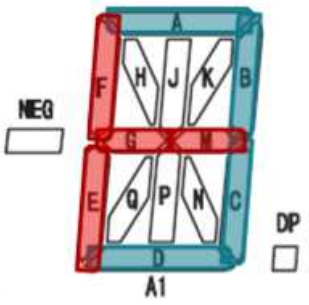
5 LCD Layout and Software Considerations

Choosing the right LCD and carefully choosing which MSP430 pins to connect to particular pins of the display can make a big difference in the ease of use of code and code efficiency. This ties into how the display muxes different areas together onto the same segment pin, in relation to what is displayed on it (for example, alphabet characters, or numbers). While deciding on a layout, taking into consideration the way that the MSP430 LCD memory is structured in different muxing modes can also help ensure efficient and easy to use software.

5.1 LCD Layout Tips

Depending on the muxing used, it may take multiple Sx pins to display an entire digit or character on the screen. Which display areas are muxed together depends on the particular LCD glass used and how it has mapped these segments to drive pins. Choosing the right LCD display can make software easier. For example, if a display has the Sx pins mapped to segments that allow it to display any numerical digit using only one or two Sx pins, that can make the code easier to write. The mapping of pins to LCD segments is found in the LCD data sheet.

Figure 13 shows some information from the data sheet for the LCD glass on the MSP-EXP430FR4133 Launchpad Development Kit, which has a 4-mux display. This display allows for showing alphanumeric characters in addition to simple digits. Looking at the pin mapping found in the display data sheet, you can see that the segments required to display any digit 0-9 take only two Sx pins – pin 1 and 2 of the LCD. To display any alphabet character, 4 segment pins are used.



PIN	1	2	3	4
COM0	A1M	A1D	A1DP	A1P
COM1	A1G	A1C	A1N	A1K
COM2	A1F	A1B	NEG	A1J
COM3	A1E	A1A	A1Q	A1H

Figure 13. 4-Mux Display Data Sheet Example

In addition to the mapping of the LCD glass itself, careful selection of which Sx segment pins on the MSP430 to connect to which LCD segment pins on the LCD glass can also have a big impact on both layout and software.

5.1.1 Hardware-Driven Layout

In a hardware-driven layout approach, the pins might simply be connected to the closest LCD-capable pins on the MSP430 so as to minimize crossings and try to layout the board in a single layer. However, this can result in layouts where the pins mapped to the MSP430 LCD memory are scattered through memory, meaning more software work and overhead when writing the code.

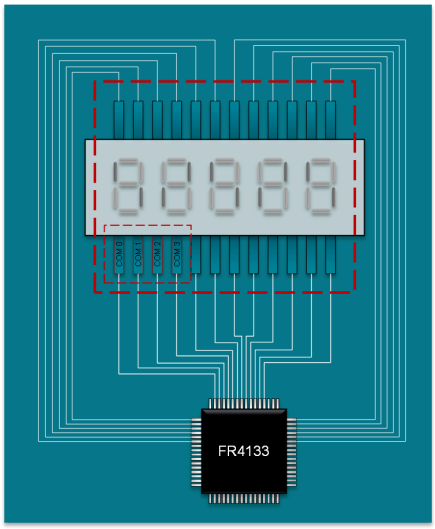
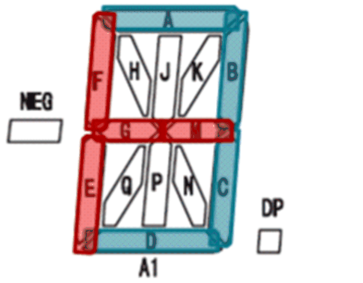


Figure 14. Example Layout Grouping LCD Lines Bus-Style in a Single Layer

5.1.2 Software-Driven Layout

Using a software-driven approach for pin selection and layout would choose to connect LCD segment pins on the LCD display to groups of LCD Sx pins on the MSP430 MCU that map adjacent areas in the LCD memory. This allows the MSP430 to set all required segments to display digits/characters using a single memory write of a byte or word. For example, for the 4-mux display shown previously in Figure 13, 2 pins control all the segments to display any digit 0-9, and 4 pins control all the segments to display any alphanumeric character. In 4-mux mode, the MSP430 LCD memory has each byte controlling 2 Sx pins. Therefore, with careful connections between the MSP430 and this LCD glass, a full digit could be set on the display with a single byte access (writing 2 pins at once), or a full alphabet character could be set on the display with a single word access (writing 4 pins at once). Figure 15 shows an example of choosing MSP430 Sx pins within the same LCDMx memory register so that all segments for both Sx pins can be set with a single byte access. It is also important to make sure that the same segments for each digit are assigned to the same ordering within the byte and are laid out in the same format in memory – this ensures that the same function call can be used no matter which digit of the display you are trying to set, saving greatly on software overhead.

However doing the layout for these connections may also be more complex depending on the application, or require a multi-layered board. On devices with the LCD_E module, any pin can be a COM pin or a segment pin helping to make layout easier, so this can help to mitigate this problem to an extent.



PIN	1	2	3	4
COM0	A1M	A1D	A1DP	A1P
COM1	A1G	A1C	A1N	A1K
COM2	A1F	A1B	NEG	A1J
COM3	A1E	A1A	A1Q	A1H

MSP430 Pin	LCD pin
S8	1
S9	2

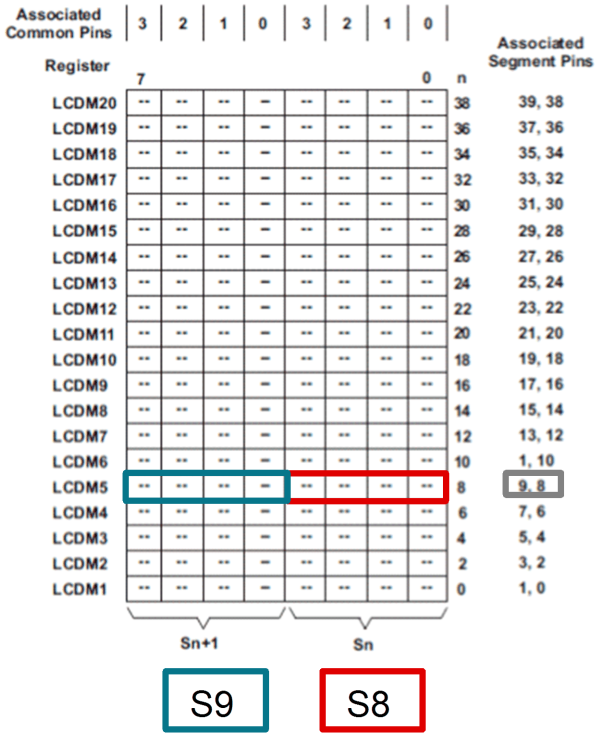


Figure 15. Software-Driven Layout Pin Selection Example

5.1.3 General Layout Rules

LCD signal lines are constantly switching to keep the image on the display, so they should be kept away from any noise-sensitive lines (like the external crystal connections). Guard rings can be used to help shield noise-sensitive lines like the crystal connections or ADC inputs from noise coupling, and having a ground plane underneath the LCD traces and guard traces can also provide shielding.

One good practice is to keep all LCD signal traces (segment and common lines) together similar to a data bus. Keeping the LCD layout in a single layer can also be helpful so that there are not LCD traces running over or under potentially sensitive traces. It is also recommended to keep the charge pump capacitor on the LCDCAP pin as close as possible to the MCU with a short trace. TI reference designs that use LCD displays (like the *Gas or Water Meter With 2 LC Sensors* reference design TIDM-LC-WATERMTR) demonstrate these good layout principles.

Figure 16 shows a portion of the water meter design showing the LCD connections. Note how the signals to the LCD at the top of the layout are grouped together, and go around the crystal oscillators X1 and X2 rather than under them to prevent noise from disturbing the crystals. The crystal circuitry also includes its own ground plane to help further shield them from noise from the LCD and other sources.

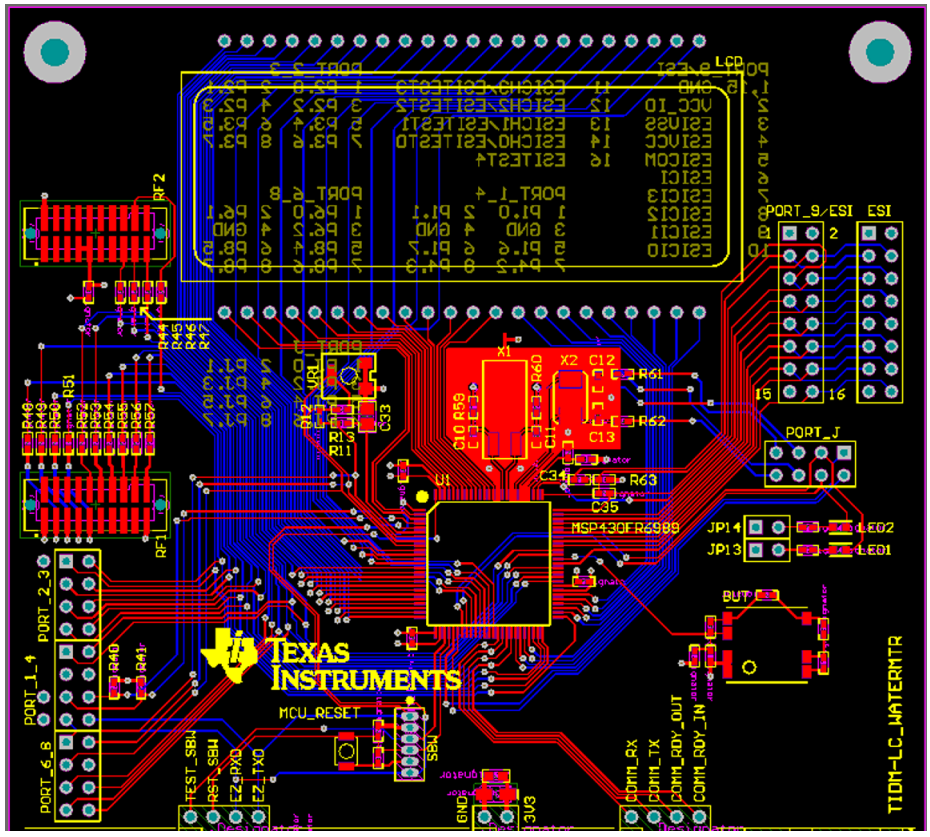


Figure 16. Portion of TIDM-LC-WATERMTR Design Showing LCD Layout

5.2 LCD Software Tips

Writing an LCD driver program can seem daunting since there are so many hardware factors that the software must handle. The LCD display glass being used and its layout and properties, the choice in connection between the LCD pins on the display and the LCD pins on the MSP430 MCU, and how the microcontroller maps the pins to display memory can make code confusing and hard to read without having the full picture and several data sheets handy. This section offers some tips to help make LCD driver software easier to use and understand, as well as create efficient LCD code. For comparison, Figure 17 shows how to display the digit "2" on the display using normal register-access methods without any of the software tips in the next few sections. Compare this to the figures and methods listed in the following sections.

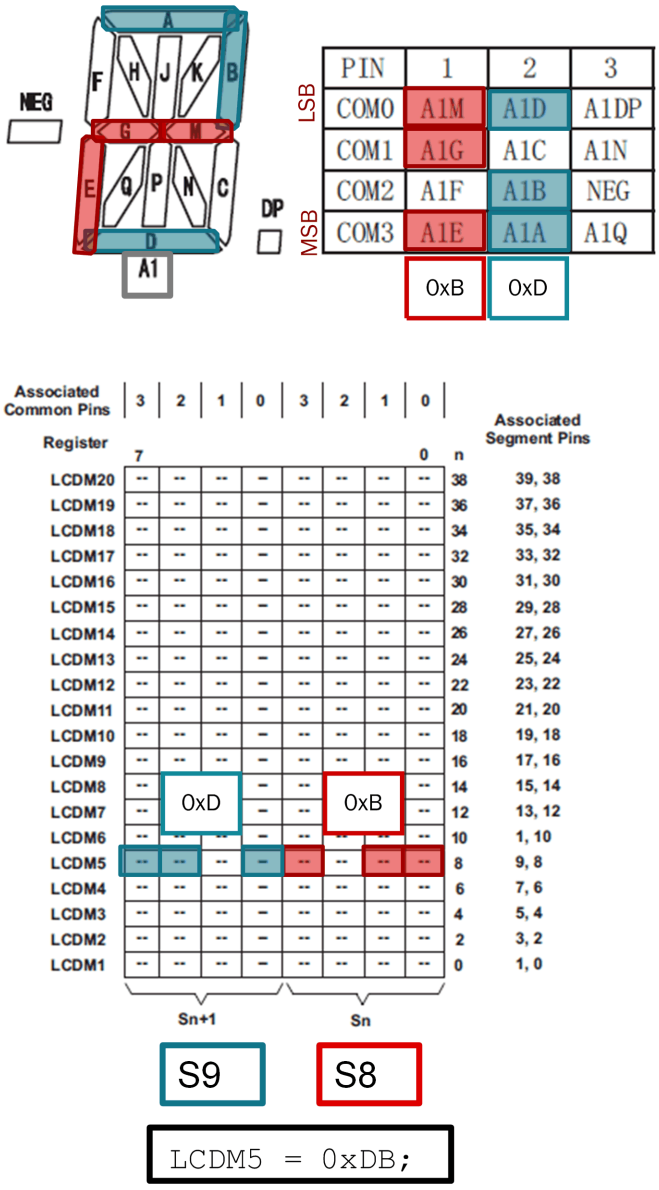


Figure 17. Example Displaying the Digit "2" Using LCD Memory (Without Special Software Techniques)

5.2.1 Create a Lookup Table

Creating a lookup table containing commonly displayed data, such as numbers, characters, or particular symbols, can make your code much easier to read. For example, if numbers will be displayed on the LCD, create a lookup table containing the values to write into the LCD memory registers to display each digit 0-9. Using the lookup table in the code snippet below, a write to display the digit 2 would look like:

```
LCDM5 = digit[2];
```

```
//lookup table for digits on MSP-EXP430FR4133 segmented LCD
const char digit[10] = {
    0xFC,  /* "0" */
    0x60,  /* "1" */
    0xDB,  /* "2" */
    0xF3,  /* "3" */
    0x67,  /* "4" */
    0xB7,  /* "5" */
    0xBF,  /* "6" */
    0xE4,  /* "7" */
    0xFF,  /* "8" */
    0xF7   /* "9" */
};

LCDM5 = digit[2]; //write a '2' on the display
```

5.2.2 Use of #defines

Because the LCD memory on the MSP430 MCU maps to particular MSP430 LCD pins, which then are connected to different pins on the LCD display, it can be difficult to know which LCD memory to write to in code in order to display a particular character in a particular space on the display. To help with this, create #defines for your LCD to let you reference the correct LCD memory by typing the name of the particular LCD display pins you are trying to set. For example, in Figure 18, the entire digit for position A1 on the LCD data sheet is set using LCDM5 register due to the pin connections on the board. Using #define A1 LCDM5, now the code can simply be written A1 = digit[2]; to write the digit 2 in position A1 on the display. This makes it much easier to write and understand the code.

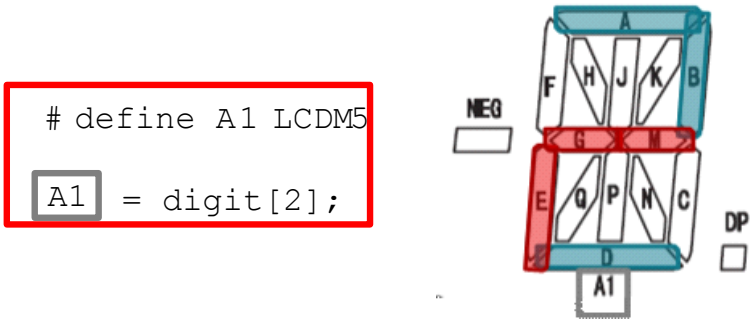


Figure 18. Using #defines for Easy-to-Read LCD Code

5.2.3 Efficient Clearing of the LCD Memory

LCD_B, LCD_C, and LCD_E modules support clearing the LCD memory using a single bit. The LCDCLRM bit can be set in these modules to clear the entire LCD memory with a single instruction – all LCD display memory registers are cleared at the next frame boundary when this is set. After the memory registers are cleared, LCDCLRM is reset, so this can be polled to see when the clear is completed. LCDCLRBM performs the same function and acts in the same way, but it clears the blinking memory registers instead.

This clearing functionality can be useful for efficiently clearing the whole screen and all memory registers – this can be useful as preparation for updating the whole display as code then must write only to memory registers that will be used in the new display data (instead of also having to manually clear other registers for unused segments).

5.2.4 Double-Buffering of the Display Buffer Using Dual Display Memory

As mentioned in Section 4.7, on LCD modules that support individual segment blinking (LCD_B, LCD_C, LCD_E) there is a blink memory that can be used as a secondary display memory when no blinking mode is selected. To select which memory (LCD memory or Blink memory) is currently being displayed simply set the LCDDISP bit to 0 or 1. The advantage of this feature is that it can be used to do an instant update of all LCD segments to display a new message or image on the screen. Changes can be made over time to the currently unused display memory without it affecting the current display output - once the full memory has been populated with the desired data the display can be changed all at once simply by toggling the LCDDISP bit.

Using this methodology, a slower MCLK can be used (sometimes required to meet low peak current consumption), or the CPU can be busy with other interrupts going on while populating this display buffer without any partial image showing up on the display.

A typical flow might be:

1. Populate the display memory that is not currently being shown with desired data
2. Toggle LCDDISP bit to change which memory is displayed
3. Go to step 1 for next image

Another feature that Dual Display Memory can provide is the ability to have the hardware automatically toggle between the two display memories – this is done by setting the blinking mode LCDBLKMODx = 3. In this mode, the LCD toggles between the memories at the blinking frequency that is configured. This can be useful for displaying a long string or text or information that does not fit on the display – part of the string can be loaded into each memory, and then this mode can be set to automatically toggle between the two memories without software intervention.

For example, on a display that can display only 6 alphanumeric characters, the message "Hello World" could be displayed by loading the LCD memory registers to display "Hello" and loading the LCD Blink memory registers to display "World", and set the LCDBLKMODx = 3, with the blink frequency configured to a slow frequency like 1 Hz to allow time for users to read the message. In this case, without any additional software intervention, the display continually shows "Hello" for 1 second and then "World" for 1 second.

5.2.5 Efficient Binary-to-BCD Conversion

When working with LCDs, it is a common task to decompose decimal/binary numbers into BCD numbers before they can actually be displayed on the LCD as digits. This typically involves rather CPU-expensive divide-by-10 and associated modulo operations. However, BCD instructions built right into every MSP430 CPU can be used to offload much of the burden of BCD conversion into hardware, resulting in more power efficient and faster executing code. On the other hand, for applications such as RTCs it may be advantageous to maintain the numbers in BCD in software throughout the application, thus foregoing the need for expensive BCD conversions entirely. The BCD-handling CPU instructions can be accessed through the C compiler intrinsic function calls __bcd_add_short() and __bcd_add_long().

```
// This function implements an efficient decimal to binary conversion.
// Note that a potential BCD overflow is not handled. In case this
// is needed, the function's return value as well as the data type
// of "Output" need to be changed from "unsigned int" to
// "unsigned long" and the intrinsics to __bcd_add_long(...).
unsigned int Dec2BCD(unsigned int Value)
{
  unsigned int i;
  unsigned int Output;

  for (i = 16, Output = 0; i; i--)          // BCD Conversion, 16-Bit
  {
    Output = __bcd_add_short(Output, Output);
    if (Value & 0x8000)
      Output = __bcd_add_short(Output, 1);
    Value <<= 1;
}
return Output;
}
```

The RTC_A, RTC_B, and RTC_C modules on some devices include a selectable BCD format for storing the calendar mode information for seconds, minutes, hours, day, month, and year in a BCD format instead of in binary (hexadecimal). When using the LCD module to display clock or calendar information, it is useful to have the RTC running in this BCD mode so that no conversion is necessary to determine the digits for the LCD display so no additional overhead. The RTC_B and RTC_C modules additionally include a BIN2BCD register that can be used to convert a 12-bit binary number to a 16 bit BCD number in hardware. This again can save on software overhead – simply write the binary number into the register and then it can be read back in a BCD format. This can be useful for efficiently converting any binary number, not just RTC calendar information, for display on the LCD.

6 Devices Without LCD Module

There are many MSP430 devices that do not have a built-in segmented LCD driver. Depending on the application, it might be necessary to use an MSP430 MCU that does not have this feature, but still need LCD display functionality. The option in this case is typically to use an LCD that has a built in LCD controller or external controller chip which can be controlled by the MSP430 via a serial communication interface (like SPI or I^2C). In some cases for very simple small displays, it might also be possible to Bit-Bang the LCD signals for a 1/2-bias display.

6.1 Bit-Banged LCD

The application note *Software Glass LCD Driver Based on MSP430 MCU* (SLAA516) provides an example of using software to generate 1/2-bias LCD waveforms on normal GPIO pins. This can be useful if you need to drive a small very simple LCD display but do not have a built-in LCD driver.

6.1.1 Bit-Banged LCD Tradeoffs

The 1/2 bias using this method is achieved by using resistor pairs and setting the pin as an input to generate the $V_{CC}/2$ state. This consumes more current than using the LCD module to drive the display, because GPIOs consume more current when held near the switching point (this is why floating pins can consume excess current). With a built-in LCD module function, the LCD pins are designed and configured for analog function so they do not incur this penalty.

In addition, the signals are being set manually by software using a timer module to do the frame timing. In the example software for the app note, a 4-mux display is driven via GPIO. Since LCD signals cannot apply a DC level to the segment, the GPIO have to toggle within each portion of the frame. So in the example the timer must wake the device 8 times in each frame to set the GPIO – there are 4 time slots because it is a 4-mux display, but the device must wake twice for each time slot to toggle the pins so that no DC voltage is applied across the time slot. The device must do these operations 8 times per frame to

maintain a constant image on the display, even when nothing on the display is changing. This means that this method has much more software overhead than using the LCD module, which can maintain the display image without CPU intervention, with software only needed when the display image should change. The additional software overhead can affect overall system throughput, can block or delay other interrupts in the system, and also means the device has to be awake for a larger percentage of the time leading to higher current consumption.

Therefore, this method should only be used for applications where these tradeoffs have been carefully considered – some applications are simple and can function adequately with this method, but for others this is too computationally intensive, consumes too much power, or a more complex display is required than this method can support.

6.2 Displays With Built-in Drivers and Serial Interface

Some LCD displays have a built in driver, or an external driver chip can be paired with the display. Many dot-matrix LCD or e-paper displays have built-in drivers like this. For these displays, typically the driver chip handles all of the LCD waveform generation and the master device simply controls the display settings and image contents using commands over a serial interface like I^2C or SPI. MSP430 microcontrollers with the USCI or eUSCI module can easily control these types of displays. However, some display drivers do not have a read-back capability, so, depending on the required display functions for the application, the MSP430 may need to store the current image contents in RAM or FRAM set aside for read/write access. It is important to remember this when doing device selection to ensure the device has enough memory if this is a requirement.

An example of a dot-matrix display with a built-in LCD driver is the Sharp LCD BoosterPack (430BOOST-SHARP96). MSP430Ware includes MSP430-GRLIB, a library for doing graphics on various dot-matrix displays including the Sharp LCD Boosterpack. This can also be used as a framework for developing a display driver for other dot-matrix displays.

7 Additional Resources

1. *MSP430F4xx Family User's Guide* (SLAU056)
2. *MSP430F5xx and MSP430F6xx Family User's Guide* (SLAU208)
3. *MSP430FR5xx and MSP430FR6xx Family User's Guide* (SLAU367)
4. *MSP430FR4xx and MSP430FR2xx Family User's Guide* (SLAU445)
5. MSP-EXP430FR4133 Launchpad (MSP-EXP430FR4133)
6. Out-of-box software (in the MSP-EXP430FR4133 Software Examples)
7. *Driving Large LCDs With LCD Peripheral of the MSP430* (SLAA272)
8. U*se of Two MSP430s to Enhance Segment Lines for Larger LCDs* (SLAA072)
9. *Software Glass LCD Driver With MSP430* (SLAA516)
10. Sharp LCD Boosterpack (430BOOST-SHARP96)
11. MSP430Ware
12. MSP430-GRLIB

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from November 21, 2014 to July 20, 2015	Page
• Changed the link destination for the MSP-EXP430FR4133 Software Examples	23

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources 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.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

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