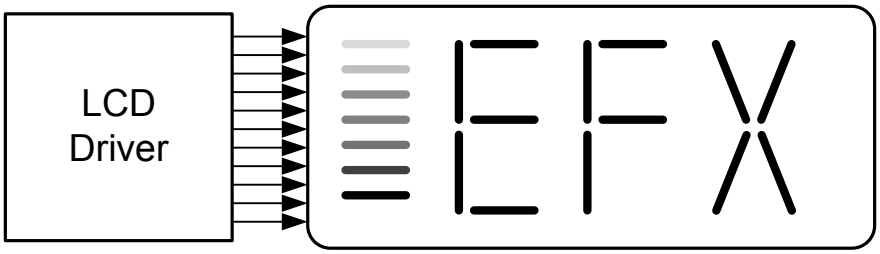
AN0057.2: EFx32 Series 2 LCD Driver

This application note provides a description of how passive segment LCD displays work and how they can be interfaced with Series 2 devices.

KEY FEATURES

- The ultra-low power LCD driver has internal bias voltage circuit to minimize external components.
- The charge pump mode enables it to provide the LCD display up to about twice the supply voltage of the device.
- The power consumption of the LCD panel itself can be lowered through the use of charge redistribution.
- The LCD driver supports autonomous animation and blinking in deep sleep without CPU intervention.



1. Device Compatibility

This application note supports multiple device families, and some functionality is different depending on the device.

EFM32 Series 2 consists of:

- EFM32PG23
- EFM32PG26
- EFM32PG28

EFR32 Series 2 consists of:

- EFR32FG23
- EFR32ZG23
- EFR32BG26
- EFR32MG26
- EFR32FG28
- EFR32SG28
- EFR32ZG28

2. Introduction to LCD Segment Displays

Segmented liquid crystal displays (LCDs) are a common way to display information. The low-power LCD driver in the EFx32 devices enables many applications to use an LCD even in energy critical systems. This document will discuss how passive LCDs work, how they are driven, and how to minimize their energy consumption.

2.1 Passive LCD Displays

Passive LCDs are made up of segments that can be individually controlled. Each segment is typically made up of two glass plates with liquid crystal in the middle. These segments can either block light or allow it to pass through depending on the voltage applied over the liquid crystal within that segment.

Each glass plate contains polarizing material that is orthogonally oriented with respect to each other. Only the back glass plane has a reflective coating. This reflective coating on one side of the glass allows ambient light to either be reflected back to the user or blocked by the polarizer closest to the reflective layer. This blocking of light occurs because the liquid crystal in one state, with no voltage applied across the liquid crystal, will change the polarization of the light to allow it to pass both polarizers. In the other state, when a voltage is applied across the liquid crystal, it will not affect the polarization, and the light is then blocked due to the orientation of the polarizers.

This document will further use the notion that a segment is "on" when it is blocking light, which is seen as a dark segment appearing on the screen and "off" when light can pass through, which is seen as the segment assuming the color of the background of the LCD. The figure below illustrates how light polarization is affected when light passes through the polarizers and liquid crystal with and without an applied voltage

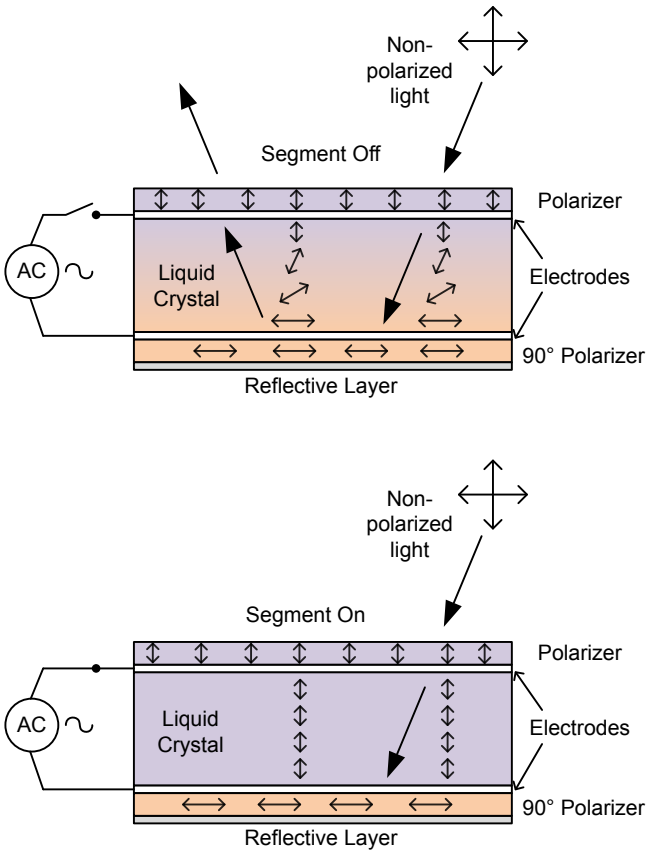


Figure 2.1. One LCD Segment with and without Voltage Applied

2.2 Driving a Display Segment

Segment LCDs do not have any internal driving circuitry. All the display pins are directly connected to each side of the corresponding segment. The simplest LCD design would consist of only one segment with two electrical connections, one for each side of the segment. The segment is turned on or off by applying a voltage across it. The electric field, or voltage, across the liquid crystal between the glass plates directly affects the polarization properties, see Figure 2.1 One LCD Segment with and without Voltage Applied on page 3.

However, applying a constant voltage across the segment is not a long term solution. After some time, the LCD segment will be affected by the DC-current passing through it, mainly through electrolysis effects on the liquid crystal and electrodes. The solution is to drive the segment with a waveform that has an average zero DC value. As long as the voltage is switched fast enough (30 - 100 Hz), it will be the RMS value of the voltage that affects the amount of polarization.

The simplest way to drive one segment of an LCD with the correct zero DC bias would be to apply two phase shifted square waveforms. By shifting the phase of one signal with respect to the other by 180 degrees, the apparent RMS voltage across the segment goes from zero to two times the voltage of the two waveforms. Since negative supplies are rarely available on microcontrollers, connecting one side of the segment to ground is not an option. This means that in microcontroller applications, using phase shifted square waveforms is the simplest solution. See the following figure.

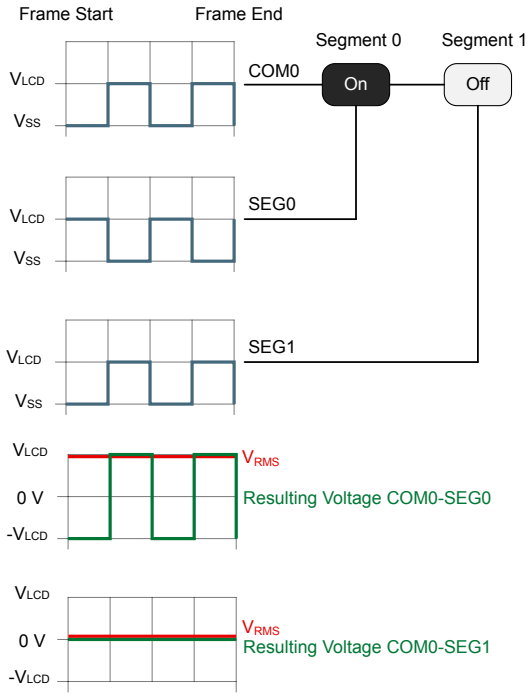


Figure 2.2. Static Driving of Two LCD Segments, One On and One Off

2.2.1 Driving Many Segments: Static Driving

Typically, LCDs contain more than one segment. Usually the segments are connected with one side common to many of the other segments, this is called the "common"-electrode. The other side of the segment has its own pin connection, the "segment"-electrode. See the following figure.

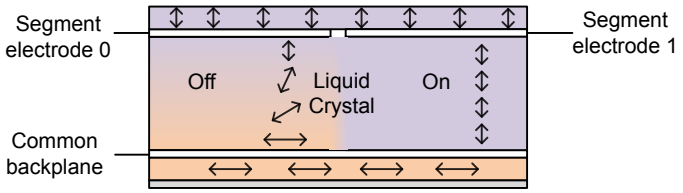


Figure 2.3. Common Backplane with Two Segments, One On and One Off

With a statically driven LCD display, there is only one common electrode and each segment has its own segment electrode. Both the common electrode and segments are driven with a square wave form. Segments that are "on" are driven with a phase shifted waveform to make the RMS voltage across these segment non-zero. See Figure 2.2 Static Driving of Two LCD Segments, One On and One Off on page 4.

2.2.2 Driving Many Segments: Multiplexed Driving

The downside to using a static driving approach is that many microcontroller pins are needed to drive a display with many segments, since each segment requires one pin. By multiplexing common and segment lines, fewer pins can be used to drive more segments. The total number of segments that can be driven is the product of the number of common lines and segment lines. As an example, if a datasheet says it can support a 4x20 LCD, this means it can support 4 common (COM) and 20 segment (SEG) electrodes, or 80 segments total. Typically maximum contrast goes down and current consumption goes up with a higher number of common lines.

The amplitude of the RMS voltage across a segment determines if a segment appears on or off. A segment with a low RMS voltage applied will seem to be off even if the voltage is non-zero. The relationship between RMS voltage across a segment and its visual properties is non-linear. This non-linearity is utilized when multiplexing several segments on the same driving pins. A segment does not need to see zero RMS voltage to be perceived as completely off by the user.

Each common line and segment line is driven with waveforms consisting of more than two voltage levels as in the static driving case. The number of voltage levels are known as "bias" levels. By carefully selecting the waveform of each segment line and common line, it is possible to make some segments "see" a low RMS voltage, while others "see" a high RMS voltage, even if the segments share either their common or segment electrode with other segments. See the figure below.

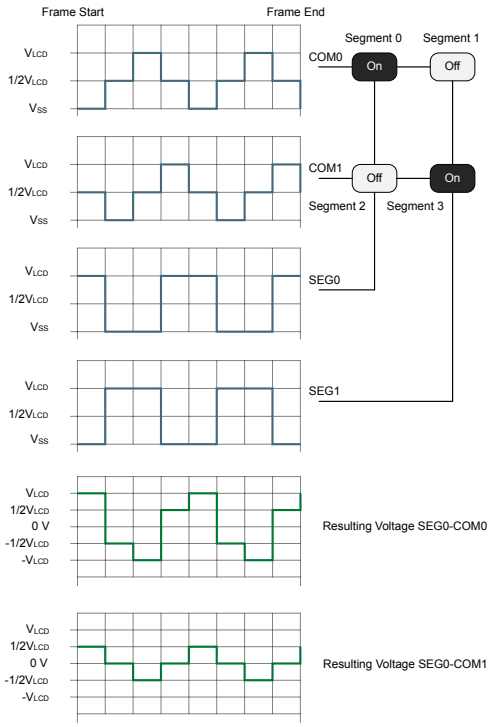


Figure 2.4. Multiplexed Driving of Four LCD Segments

In the figure above there are two segment lines and two common lines. The bias is 1/2, which means there are 3 different voltage levels possible. The combination of the waveform on COM0 and SEG0 makes the segment between SEG0 and COM0 turn on, while the segment between SEG0 and COM1 is off. Notice the difference in RMS voltage across the two segments.

For a multiplexed display, the difference in RMS voltage between segments that are on and off will be smaller than for a statically driven display. This means that the visual quality and appearance, such as contrast and viewing angle, must be more carefully calibrated for a multiplexed display. The contrast can be calibrated in software by adjusting the bias voltage levels within the LCD driver. Viewing angle and appearance under different lighting conditions should be considered when selecting the actual display.

3. EFx32 LCD Driver

This chapter explains some of the features and possibilities of the EFx32 built-in LCD driving peripheral. For a full set of features and limitations, please see the reference manual for the specific device family.

3.1 Frame Rates

Frame rates for LCDs are typically set between 30 and 100 frames per second. This range is typically used since it prevents flickering of the display or low contrast in the segments. The LCD peripheral driver allows for configurable frame rates by including a frame rate divider in the `LCD_FRAMERATE` register that is able to determine the number of prescaled clocks per phase. Refer to the reference manual for more information on frame rates.

3.2 Multiplexing

The Series 2 LCD driver supports multiplexing and static drive settings. These settings are handled in the `LCD_DISPCTRL` register. The multiplex setting determines the number of LCD COM lines that are enabled. The higher number of COM lines that are used, the more the current consumption of the peripheral increases.

The four multiplexing settings are static, duplex, triplex, and quadruplex. When static multiplexing is used, the LCD output is enabled on COM0, when duplex is used, COM0-COM1 are enabled, when triplex is used, COM0-COM2 are enabled, and when quadruplex is used, COM0-COM3 are enabled. Refer to the device specific reference manual for more information on LCD multiplexing.

3.3 Bias

Drive bias settings describe the number of voltage levels that are used to drive segments on the LCD. Higher bias settings are needed when more segments are driven on each COM line. Selecting the correct bias is necessary to provide the waveform generator with enough voltage levels to produce the necessary waveform for the chosen number of common lines. Four bias settings, static, half, third, and fourth bias, are supported by the EFx32 Series 2 LCD driver. Bias settings are described as 1/(N-1), where N represents the number of voltage levels. For example, a half bias setting would indicate that 3 voltage levels are present. These bias modes can be set in the `LCD_DISPCTRL` register.

Note that for some multiplex configurations, contrast actually goes down with an increased number of bias levels. In general a higher number of bias levels also means higher energy consumption, so for a given number of multiplexed common lines, the lowest possible number of bias levels should be chosen. Depending on the LCD, contrast might be acceptable even with a lower number of bias levels.

3.4 Waveform Selection

Type A, or regular waveform outputs, and Type B, or low power waveform outputs, are selectable wave settings in the `LCD_DISPTRL` register. The low power waveforms are able to achieve zero DC bias over several frames. This means, when these waveforms are selected, the DC bias across LCD segments in a single fram is non-zero. This is often a recommended solution for most LCDs.

Figure 3.1 Normal and Low Power Waveform Across One COM Line on page 8 shows the difference between a type A and type B waveform. Typically, the normal type A waveforms include more changes between voltage levels within a frame when compared to the low power type B wavefoms. If flickering or contrast problems occur are observed when using a low power waveform setting, normal mode can be selected instead. Refer to the device reference manual for more details and examples of waveform and bias settings.

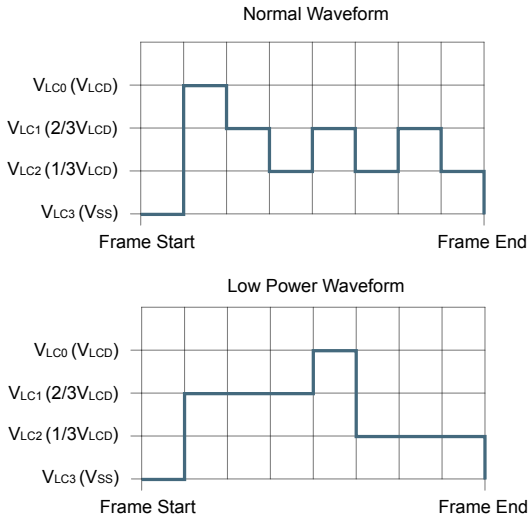


Figure 3.1. Normal and Low Power Waveform Across One COM Line

3.5 Contrast

As discussed previously, a non-zero RMS voltage is present across all segments, even off segments, in a multiplexed display. This means that adjusting the contrast of the LCD is usually necessary when using multiplexed displays. Contrast adjustment is important because it helps the user distinguish between on and off segments, even with this non-zero RMS voltage. The user must determine at which threshold voltages the segments can be perceived as on or off. The difference between V_{on} and V_{off} is essentially a measurement of the possible contrast. For the static driven display, the contrast is essentially infinite. Additionally, contrast varies among LCDs and changes with temperature, voltage, and viewing angle. More information on contrast compensation may be available in the documentation for the LCD.

See Figure 3.2 Relative Transmission of Light Through the LCD for Changes in the Applied RMS Voltage on page 9 for an illustration of the non-linear nature of the liquid crystal material with respect to applied voltage and transmission of light without changing polarization.

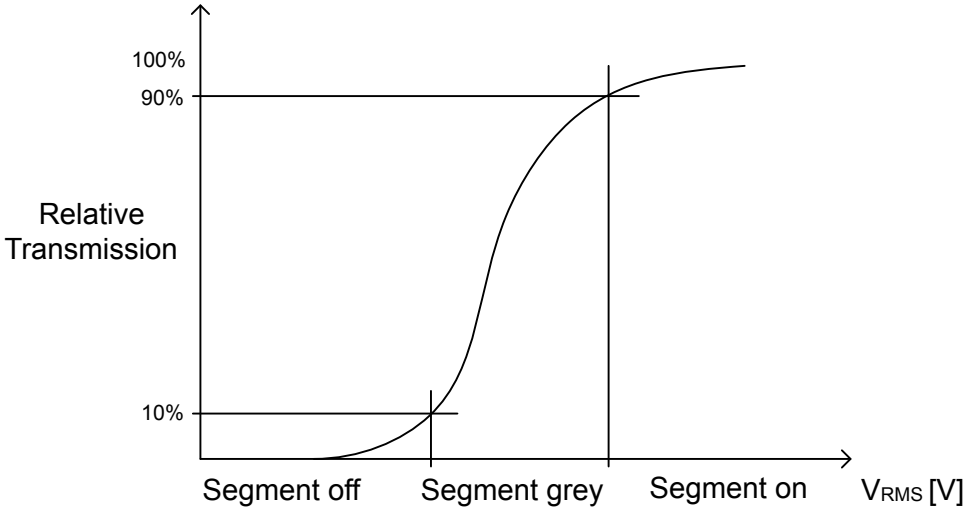


Figure 3.2. Relative Transmission of Light Through the LCD for Changes in the Applied RMS Voltage

3.5.1 Charge Pump Mode

Some LCDs require higher voltage waveforms for operation. The Series 2 LCD peripheral supports charge pump mode to generate a V_{LCD} voltage up to 1.95 * V_{DD}. The charge pump can charge to the range of 2.4 V - 3.6 V by configuring the `LCD_BIASCTRL.VLCD` field. Additionally, the Series 2 LCD driver runs in step down mode by default, so charge pump mode must be selected in the `LCD_BIASCTRL.MODE` field.

Charge pump mode is used when the contrast levels are set to the maximum possible values and the contrast of on segments on the LCD appear dim. Reduced contrast and even perceived blanking of the display is also a problem at low temperatures and can be resolved in software by enabling charge pump mode to drive V_{LCD} at a higher voltage level. Refer to the device reference manual for more information on charge pump mode.

3.6 Animation and Blinking

The EFx32 LCD driver includes special features to enable animation and blinking of specific segments without any software intervention. This is useful for displaying continuous animation to signal that a device is alive, without the need to wake up from deep sleep to update the segments.

The animation feature is available on segments 0 to 7, or segments 8 to 15, multiplexed with LCD_COM0. The animation is implemented as two programmable 8-bit registers that are shifted left or right every other animation state for a total of 16 states. The animation state changes with each frame counter event. See the device reference manual for more information on the animation feature and its interface. The animation registers and shift operation are shown in the following image.

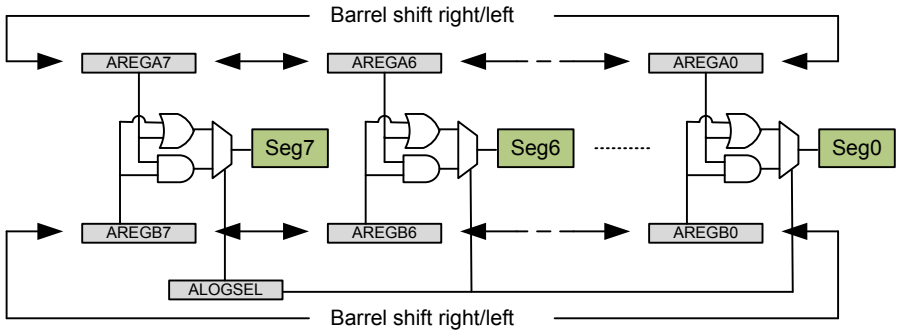


Figure 3.3. Animation Function with Barrel Shift and Logic Operations

3.7 LCD Interrupts

The LCD driver on Series 2 devices can generate interrupts from three different sources. There is a display counter interrupt that can be used to indicate when the display count reaches zero, as well as an interrupt that can be used to synchronize data updates. Similar to Series 1 devices, there is also a frame counter interrupt that generates an interrupt with each LCD frame counter event, which is set up separately. Refer to the reference manual for each device family for more information on LCD interrupts.

3.8 Minimize Energy Consumption

The LCD driver is itself a low power peripheral that can operate in deep sleep mode (EM2) from a low frequency oscillator. When connecting a display to the EFx32, the capacitive load presented by the LCD results in increased energy consumption. This can be minimized by optimizing a few parameters.

After the multiplex, bias, and waveform settings are configured, the display refresh rate should be configured to the lowest possible value that does not cause flickering by setting the frame rate division and prescale values.

Charge pump mode should not be used unless absolutely necessary. Instead, contrast should be adjusted to counteract lower supply voltage or temperatures. Remember that charge pump mode significantly increases current consumption.

In addition, the power consumption of the LCD panel itself can be lowered through the use of charge redistribution. When charge redistribution is in use, all segments are briefly shorted together to the same mid-voltage driver, allowing low segments to be partially charged by the energy in high segments instead of using additional energy from the power supply. Charge redistribution should be set to less than 5% in the LCD_DISPCTRL.CHGRDST field to prevent any reduction in the LCD pin RMS voltage. See the reference manual for for how to calculate and set charge redistribution duty cycle.

If charge redistribution is not used, a larger CMU prescaling value is recommended to minimize power consumed by the LCD block itself. Note that disabling charge redistribution will always result in higher system power consumption. Charge redistribution is on by default, but it can be disabled by setting LCD_DISPCTRL.CHGRDST to zero.

4. Software Examples

Software examples are available in Silicon Labs' platform_applications repository on Github. These examples show a variety of low energy applications using the LCD peripheral on EFx32 Series 2 devices. These examples include using the LCD to display data from the LC sensor, RHT sensor, timers, LDMA, animation features, and a low power example. Refer to the readme of each example for more details.

4.1 Porting the LCD Driver to other Displays

Porting or writing an LCD driver for the specific display depends on understanding the COM and Segment components of the segment LCD being used and the LCD peripheral available on the Series 2 device. For example, 4-plexed displays are supported for EFR32xG23 and EFM32PG23 devices, since 4 COMs are available, while 8-plexed displays are supported for EFR32xG26 and EFM32PG26 devices, and EFR32xG28 and EFM32PG28 devices, since 8 COMs are available.

Most segment LCDs are able to display information through individual symbols, 7-segment displays, and 14 -segment alphanumeric displays. Examples of individual symbol segments are a battery symbol, a Celsius temperature indication, or decimal points. These individual symbols are shown in the kit schematics and segment maps. Figure 4.3 EFM32PG23 Segment LCD Map on page 13 shows P2, P3, P4, P5, and P6, which are a few examples of individual symbols on the EFR32xG23 and EFM32PG23 kits.

7-segment displays are built from 7 individual segments that can be turned on or off to display numbers from 0 to 9. They are named from A to G clockwise around the display, and ending with G as the segment in the middle of the display.

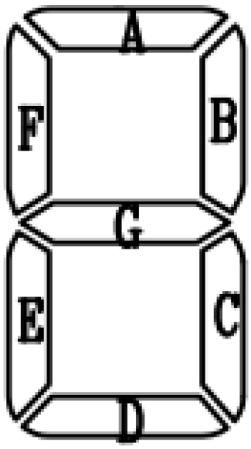


Figure 4.1. Seven Segment Digit with Typical Naming of Segments

14-segment alphanumeric letters, are built from 14 individual segments that can be turned on or off to display numbers from 0 to 9, upper and lowercase letters, and symbols. They are named clockwise starting with A from the top of the display. It is important to note that these displays require more pins in order to be used.

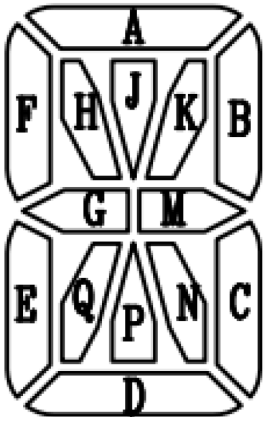
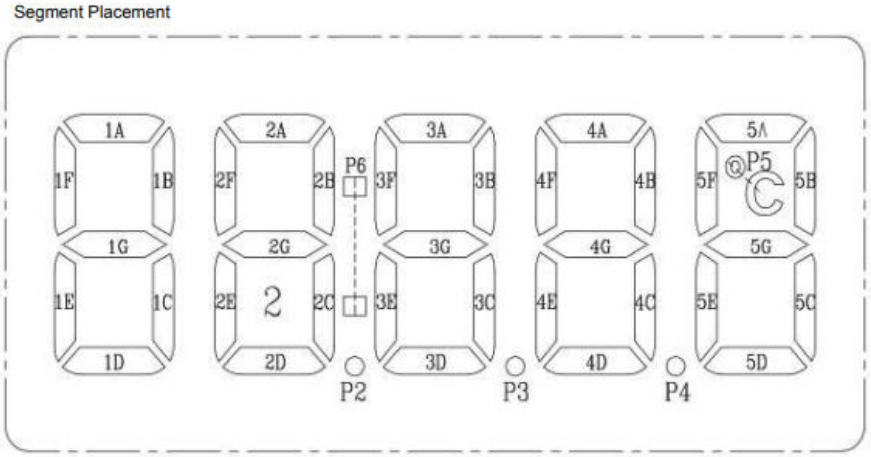


Figure 4.2. Fourteen Segment Alphanumeric Letter with Typical Naming of Segments

4.1.1 Understanding the LCD Segment Map

Segment maps contain a multiplexing table that helps determine what common and segment lines (also reffered to as pins) are connected to each individual LCD segments. Typically, the schematic for each kit that features an LCD will show a segment placement figure and segment names table that can be used to write a multiplexing table. Figure 4.3 EFM32PG23 Segment LCD Map on page 13 and Figure 4.4 EFM32PG26 and EFM32PG28 Segment LCD Map on page 13 show the segment LCD pin assignments on the EFM32PG23 and EFM32PG26/EFM32PG28 pro kits.



Segment Names

PIN	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
COM1	1D	1C	2D	P2	3D	P3	4D	4C	P4	5D	COM1	--	--	--	--	--	--	--	--	--
COM2	1E	1G	2E	2C	3E	3C	4E	4G	5E	5C	--	COM2	--	--	--	--	--	--	--	--
COM3	1F	1B	2F	2G	3F	3G	4F	4B	5G	5B	--	--	COM3	--	--	--	--	--	--	--
COM4	1A	P6	2A	2B	3A	3B	4A	P5	5F	5A	--	--	--	COM4	--	--	--	--	--	--

Figure 4.3. EFM32PG23 Segment LCD Map

PIN	1	2	3	4	5	6	7	8	9	10	11	12	13	14
——	S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13
COM0	DP2	1 E	1 D	2 E	2 D	3 E	3 D	4 E	4 D	DP5	5 D	DP6	6 D	7 E
COM1	DP4	1 Q	1 N	2 Q	2 N	3 Q	3 N	4 Q	4 N	5 E	5 N	6 E	6 N	7 Q
COM2	DP3	1 P	1 C	2 P	2 C	3 P	3 C	4 P	4 C	5 Q	5 C	6 Q	6 C	7 P
COM3	COL3	1 G	1 M	2 G	2 M	3 G	3 M	4 G	4 M	5 P	5 M	6 P	6 M	7 G
COM4	MINUS	1 F	1 J	2 F	2 J	3 F	3 J	4 F	4 J	5 G	5 J	6 G	6 J	7 F
COM5	PAD1	1 H	1 K	2 H	2 K	3 H	3 K	4 H	4 K	5 F	5 K	6 F	6 K	7 H
COM6	GEK	1 A	1 B	2 A	2 B	3 A	3 B	4 A	4 B	5 H	5 B	6 H	6 B	7 A
COM7	A7	A6	A5	A4	A3	A2	A1	A0	EFM	5 A	COL5	6 A	ANT	BAT

PIN	15	16	17	18	19	20	21	22	23	24	25	26	27	28
——	S14	S15	S16	S17	S18	S19	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0
COM0	7 D	11 A	10 A	9 A	8 A	EM2								COM0
COM1	7 N	11 F	10 F	9 F	8 F	EM4							COM1	
COM2	7 C	11 B	10 B	9 B	8 B	COL10						COM2		
COM3	7 M	11 G	10 G	9 G	8 G	DP10					COM3			
COM4	7 J	11 E	10 E	9 E	8 E	PAD0				COM4				
COM5	7 K	11 C	10 C	9 C	8 C	EM3			COM5					
COM6	7 B	11 D	10 D	9 D	8 D	EM1		COM6						
COM7	°C	°F	B1	B0	B2	EM0	COM7							

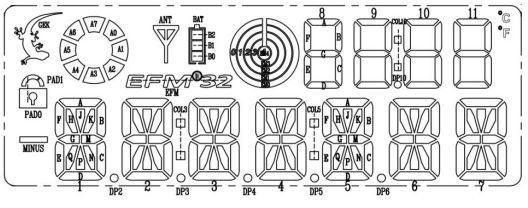


Figure 4.4. EFM32PG26 and EFM32PG28 Segment LCD Map

The multiplexing table is defined within the Platform > Board Drivers > Segmented LCD software component, within the Simplicity SDK. Specifically, pin mapping definitions for specific development boards are found in `sl_segmentlcd_pin_config.h`, included in "sisdk/hardware/driver/segment_lcd/inc/brdXXXXX" directory within the project when the software component is installed, and varies between different kits. Segment combinations which form text, numbers, or symbols are defined in `sl_segmentlcd_config.h`, again, included with the Segment LCD software component and found in the projects "config" directory. Below is a section of code from the config file

for the EFM32PG23 Pro Kit. It is helpful to refer to Figure 4.3 EFM32PG23 Segment LCD Map on page 13, which can also be found in the kit schematic, while looking at this code snippet to understand how the segments are mapped.

```
// Multiplexing table: See board schematic for more details
#define SL_SEGMENT_LCD_EFM_DISPLAY_DEF {                                  \
    .number      = {                                                      \
      { /* #5 location : Segments 5A/5B/5C/5D/5E/5F/5G */                 \
        .com[0] = SL_SEGMENT_LCD_COM_C03, .com[1] = SL_SEGMENT_LCD_COM_C02, \
        .com[2] = SL_SEGMENT_LCD_COM_C01, .com[3] = SL_SEGMENT_LCD_COM_C00, \
        .bit[0] = SL_SEGMENT_LCD_SEG_S09, .bit[1] = SL_SEGMENT_LCD_SEG_S09, \
        .bit[2] = SL_SEGMENT_LCD_SEG_S09, .bit[3] = SL_SEGMENT_LCD_SEG_S09, \
        .com[4] = SL_SEGMENT_LCD_COM_C01, .com[5] = SL_SEGMENT_LCD_COM_C03, \
        .com[6] = SL_SEGMENT_LCD_COM_C02,                                 \
        .bit[4] = SL_SEGMENT_LCD_SEG_S08, .bit[5] = SL_SEGMENT_LCD_SEG_S08, \
        .bit[6] = SL_SEGMENT_LCD_SEG_S08,                                 \
      },
```

This code section shows the fifth 7-segment group, specifically, the one that has the Celsius symbol within the 7-segment group, that is available on the EFM32PG23 pro kit. This means the code above contains mapping information for LCD segments 5A through 5G. The mapping of each LCD segment is defined by a COM line and a PIN, for example, the first segment in this group would be 5A. This corresponds to `.com[0]` and `.bit[0]` in the `SL_SEGMENT_LCD_EFM_DISPLAY_DEF` define in the code above. This means segment 5A is mapped to COM3 and SEG09 (defined in sl_segmentlcd_pin_config.h as pin 10).
Ensure that you are mapping the LCD segment to the correct pins by refering to the segment LCD signal connections section of the schematic for the kit.

Individual segments, such as segments P2-P6 in figure, are simply mapped by defining each COM and SEG lines. The code section below shows the define for the degrees Celsius symbol on the EFM32PG23 pro kit.

```
#define SL_LCD_SYMBOL_DEGC_COM  SL_SEGMENT_LCD_COM_C03
#define SL_LCD_SYMBOL_DEGC_SEG  SL_SEGMENT_LCD_SEG_S07
```

4.1.2 Porting the LCD Segment Map

The LCD segment map can be ported to other 4-plexed or 8-plexed displays. The recommended first step in this process is to start by porting any individual symbol segments that may be present on the display. As mentioned in the previous section, these individual segments have their own common and segment line defines at the start of the `sl_segmentlcd_config.h` file. Verify that the individual segments are correctly mapped by turning on only those segments and checking the display. The most common problems that can occur while porting the segment map include reversal of the common and segment lines, errors in the LCD module documentation, and PCB schematic or layout errors.

The segment digits should be ported next, if there are any. Again, `.com[0]` and `.bit[0]` refer to the common and segment line of the first segment (segment A) in the segment digit, `.com[1]` and `.bit[1]` to segment B, and so on. These are most often named A-G in alphabetic order, clockwise around the 7-segment digit, starting at the top, as shown in Figure 4.1 Seven Segment Digit with Typical Naming of Segments on page 12.

If the display contains 14-segment displays, these can be ported in the same way as the 7-segment digits. The labeling of the segments is typically clockwise starting from the top. Manufacturers have variations in the letters used to represent each segment on the display. Refer to the specific LCD datasheet for naming conventions.

5. Revision History

Revision 1.2

April 2025

- Adds EFR32xG26 and EFM32PG26 support to document.
- Updates to 4.1.1 Understanding the LCD Segment Map for Segment LCD software component.

Revision 1.1

June 2023

- Adds EFR32xG28 and EFM32PG28 support to document.
- Minor edits to 4.1 Porting the LCD Driver to other Displays.

Revision 1.0

September 2022

- Initial Revision

Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support & Community
www.silabs.com/community

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

www.silabs.com