Atmel AVR1025: How to Specify a Custom LCD for XMEGA B Devices

Features

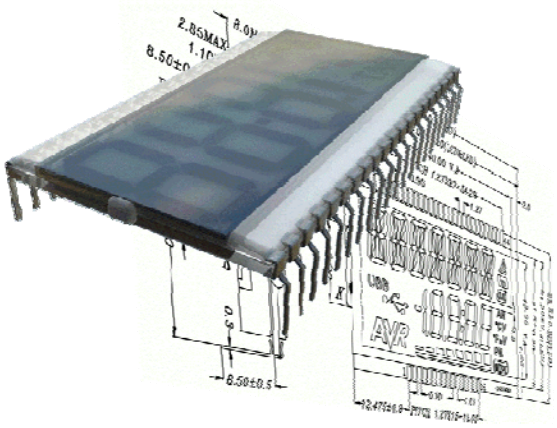
- **Segmented LCD glass (2.5V – 3.5V)**
- **Up to four common lines × 40 segment lines**
- **ASCII character mapping**
- **Automatic segment blinking**
- **Backlight**

1 Introduction

The ultra-low-power LCD controller in Atmel® AVR® XMEGA® B microcontrollers supports displays with up to 160 segments, and delivers built-in contrast control, ASCII character mapping, programmable segment blinking, efficient interrupt process and swap mode.

The ASCII character mapping built-in the LCD controller allows setting and clearing 7-, 14- or 16-segment characters simply by writing the corresponding ASCII code. Programmable segment blinking allows blinking up to eight autonomous segments with selectable blink rate. This makes AVR XMEGA B devices ideal for clocks, alarms or anything where some type of dynamic indicator is needed.

Since the LCD controller of AVR XMEGA B microcontrollers have features that reduces design complexity, this application note helps you in the design of your own LCD panel to enable these benefits. An LCD design well adapted to the controller characteristics will simplify source code and will reduce the time spent in active mode.



8-bit Atmel Microcontrollers

Application Note

Rev. 8467A-AVR-11/12

2 LCD design

The LCD panel specification is the design of each segment within the visible area of the glass.

100-pins AVR XMEGA B microcontrollers can drive up to 160 segments organized in a 4 × 40-matrix (four common lines by 40 segment lines) and reduced configurations are possible (y common lines by × segment lines, refer to XMEGA B microcontroller manual). In the same way, 64-pin AVR XMEGA B microcontrollers can drive up to 100 segments organized in a 4 × 25-matrix (four common lines by 25 segment lines).

The design strategy will be to optimize the filling-in of the final matrix (Table 2-1) taking into account all LCD controller features and constraints.

Table 2-1. Type of matrix to fill-in.

	SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	…	SEG x-2	SEG x-1	SEG x
COM0							…			
COM1							…			
…	…	…	…	…	…	…	…	…	…	…
COM y							…			

3 Group definition

First it is recommended to create groups of segments. A list of usual groups is given below for guidelines, but the user's application can reveal other groups.

An organization into a hierarchy will be used for a better optimization of resources.

1. Blinking segments.
2. Segmented characters group or lines of characters.
3. Organized segments for animation.
4. Standalone segments.

3.1 Blinking segments

Up to eight segments can be configured to automatically blink. These segments must be connected to SEG1 and/or SEG0. The automatic blinking remains available in idle mode, in power-save mode and in extended standby mode.

All segments of this group will be named: **Xn**.

NOTE

The blink rate frequency is programmable: 0.5, 1, 2, or 4Hz. If there is no segment to blink while the automatic blinking function is enabled, then the blinking is applied to all enabled segments in the matrix.

3.2 Lines of characters

The Atmel AVR XMEGA B LCD controller can automatically handle ASCII characters. Instead of setting and clearing segments of a digit, the user enters the ASCII code and the digit decoder updates the segment values in the display memory.

All segments of this group will be named: **Xn-y**.

- "**n**" is the digit number (preferably incremented from left to right)

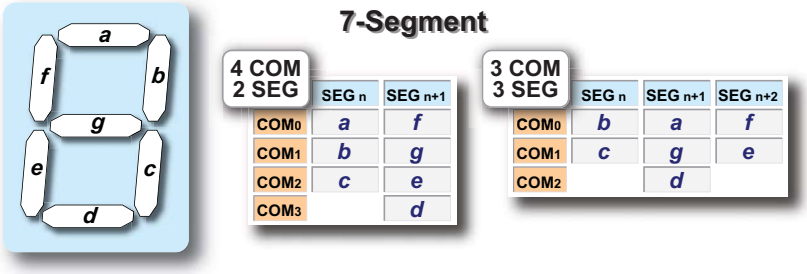
- "**y**" is the segment of the character, refer to Figure 3-1 up to Figure 3-3

Up to four types of character mapping are supported; each of them is COM0-aligned.

3.2.1 7-segment characters

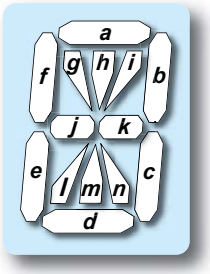
Two sub-matrixes are available:

Figure 3-1. Sub-matrixes for 7-segment character.



3.2.2 14-segment characters

Figure 3-2. Sub-matrix for 14-segment character.

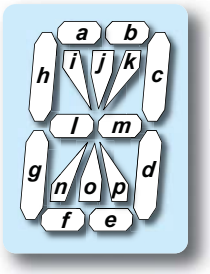


14-Segment

4 COM 4 SEG	SEG n	SEG n+1	SEG n+2	SEG n+3
COM0	h		a	g
COM1	i	f	b	j
COM2	k	e	c	l
COM3	n	d		m

3.2.3 16-segment characters

Figure 3-3. Sub-matrix for 16-segment characters.



16-Segment

3 COM 6 SEG	SEG n	SEG n+1	SEG n+2	SEG n+3	SEG n+4	SEG n+5
COM0	h	a	b	c	j	
COM1	g	i	k	d	o	
COM2	n	l	m	e	p	f

3.3 Segments for animation

Progress bar, battery indicator and any type of dynamic indicator can be grouped to reduce register access when the program needs to enable or disable them (optimized function driver).

All segments of this group will be named: **Xn-y**.

- "**n**" is the group number
- "**y**" refers to the segment within the group

3.4 Standalone segments

In this group there are all the segments considered as icons. Each of them is independent from the others.

All segments of this group will be named: **Xn**.

4 Segments location

For LCD panels that do not use the full range of segments that the device can manage, it is possible to mask some of the unused SEG pins and use them as standard GPIO pins. To free up as many SEG pins as possible it is preferably to compact the matrix, using the maximum of COM.

In the LCD controller, registers data are laid by COM (refer to XMEGA B microcontroller manual). A judicious assignment of segments by COM allows fast access reducing management time and saving power consumption.

4.1 Blinking segments location

Segments connected to SEG1 and/or SEG0 (maximum eight segments) can blink automatically.

- If less than the full range of these segments is used, assignment by COM is suggested (example for four segments: COM0/SEG0, COM0/SEG1, COM1/SEG0 and COM1/SEG1)
- If you want to blink more than eight segments on your display, the excess segments will have to be manually managed for blinking

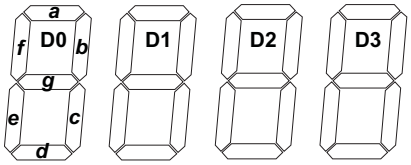
4.2 Characters location

Different lines of characters can coexist on the same LCD panel. Each line must be made of identical digits (ex: 7-segment digit for numbers), but a line can be made with another type of digits than its neighbor.

The main goal is to take advantage of the powerful of the Digit Decoder. To have the same character scan order than in a C-string, it is preferable to order the digits of the line from left to right (see Atmel AVR1618: ATxmegaB ASCII Character Mapping).

- The suggested method is to begin by the line of characters that have the bigger sub-matrix (ex: 14-segment sub-matrix)
    - Place the sub-matrix (Figure 3-1 up to Figure 3-3) of the first character as close as possible from the first free SEG column (SEG0 and SEG1 columns should be used for blinking segments, so the first SEG column will be SEG2)
    - Then place all the other sub-matrixes consecutively to build the line of characters
    - Verify if there is no overflow
    - Note the unused segments
- Continue with the next line of characters using the same method as above

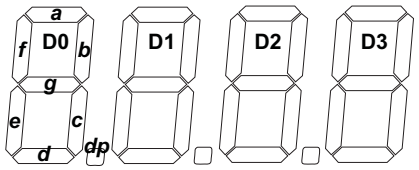
Table 4-1. Example: Four 7-Segment Characters (4COM/2SEG).



	…	SEG n	SEG n+1	SEG n+2	SEG n+3	SEG n+4	SEG n+5	SEG n+6	SEG n+7	…
COM0	…	D0-a	D0-f	D1-a	D1-f	D2-a	D2-f	D3-a	D3-f	…
COM1	…	D0-b	D0-g	D1-b	D1-g	D2-b	D2-g	D3-b	D3-g	…
COM2	…	D0-c	D0-e	D1-c	D1-e	D2-c	D2-e	D3-c	D3-e	…
COM3	…	/	D0-d	/	D1-d	/	D2-d	/	D3-d	…

- Dot points (‘DP’) are often used within numeric lines. ‘DP’ cannot be managed by the Digit Decoder. For compactness, it is interesting to associate a ‘DP’ with its left digit. A ‘DP’ can be assigned to a hole of 7-segment matrix (ex: COM3/SEGn of 4COM/2SEG matrix)

  Table 4-2. Example: Four 7-Segment Characters with Dots (4COM/2SEG).



	…	SEG n	SEG n+1	SEG n+2	SEG n+3	SEG n+4	SEG n+5	SEG n+6	SEG n+7	…
COM0	…	D0-a	D0-f	D1-a	D1-f	D2-a	D2-f	D3-a	D3-f	…
COM1	…	D0-b	D0-g	D1-b	D1-g	D2-b	D2-g	D3-b	D3-g	…
COM2	…	D0-c	D0-e	D1-c	D1-e	D2-c	D2-e	D3-c	D3-e	…
COM3	…	D0-dp	D0-d	D1-dp	D1-d	D2-dp	D2-d	/	D3-d	…

- Sometimes, a numeric line begins by 1 or -1 (else 0 but it is not displayed). Figure 4-1 proposes some solutions to integrate ±1 in both 14-segment sub-matrixes using only one SEG line. Unfortunately with this implementation this specific digit cannot be managed by the Digit Decoder

Figure 4-1. ±1 integrated in sub-matrixes for 14-segment character.

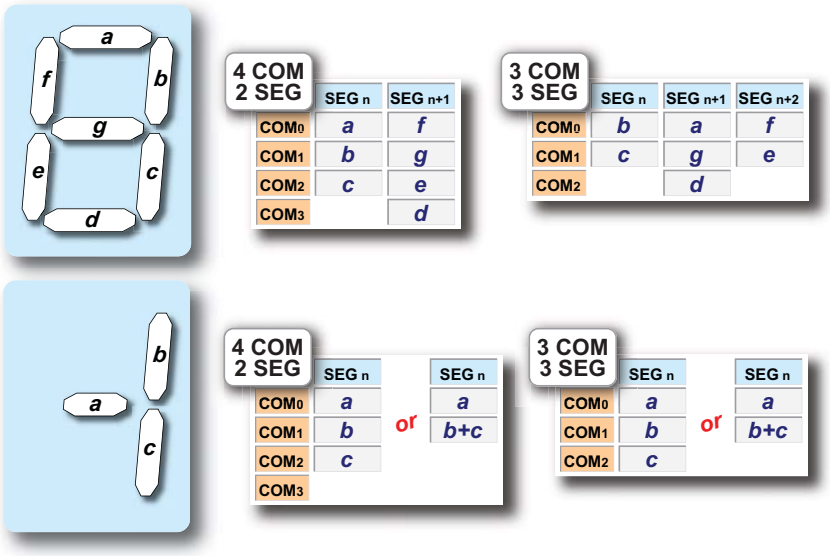
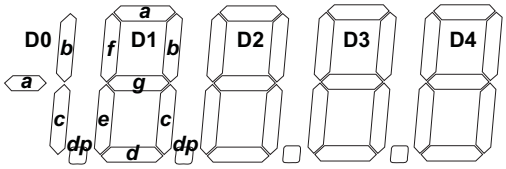


Table 4-3. Example: ±1 and 4 × 7-segment characters with dots (4COM/2SEG).



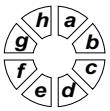
	…	SEG n	SEG n+1	SEG n+2	SEG n+3	SEG n+4	SEG n+5	SEG n+6	SEG n+7	SEG n+8	…
COM0	…	D0-a	D1-f	D1-f	D2-a	D2-f	D3-a	D3-f	D4-a	D4-f	…
COM1	…	D0-b	D1-g	D1-g	D2-b	D2-g	D3-b	D3-g	D4-b	D4-g	…
COM2	…	D0-c	D1-e	D1-e	D2-c	D2-e	D3-c	D3-e	D4-c	D4-e	…
COM3	…	D0-dp	D1-dp	D1-d	D2-dp	D2-d	D3-dp	D3-d	/	D4-d	…

4.3 Segments for animation location

To animate segments, it is recommended to create a software driver. To optimize the animation function, the segments can be placed in a single data register. By reading-modifying-writing only one register, the animation can go quickly from one step to the following one. Note that the segments are driven by only one COM.

A random location for these segments will also work well but the animation function will take more time.

Table 4-4. Example: Eight segments for animation.



C0

…	SEG 8n	SEG 8n+1	SEG 8n+2	SEG 8n+3	SEG 8n+4	SEG 8n+5	SEG 8n+6	SEG 8n+7	…	
…	…	…	…	…	…	…	…	…	…	…
COMy	…	C0-a	C0-b	C0-c	C0-d	C0-e	C0-f	C0-g	C0-h	…
…	…	…	…	…	…	…	…	…	…	…

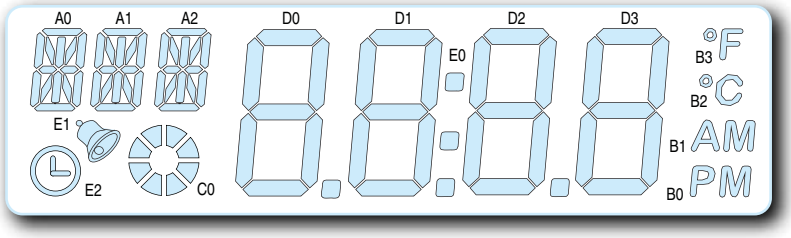
4.4 Standalone segments location

A standalone segment can be located wherever there is a ‘hole’. For these segments there is only one rule: fill-in the full matrix reducing as much as possible the number of SEG lines used to free up GPIO pins.

5 Example of implementation

This example of LCD panel for alarm clock fits with a 64-pins Atmel AVR XMEGA B microcontroller that can drive up to 100 segments organized in a 4 × 25-matrix (four common lines by 25 segment lines).

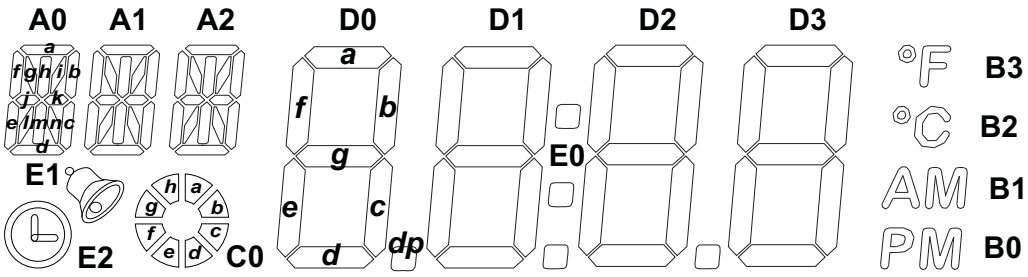
Figure 5-1. LCD panel example.



Groups:

1. Autonomous blinking segments:
   - **E0**: colon between numeric digits (blinking for seconds display)
   - **E1**: alarm (blinking if setting)
   - **E2**: stopwatch mode
2. Segmented characters group or lines of characters:
   - **A[0:2]**: 14-segment (4COM/4SEG) alphanumeric digits (day, month or command)
   - **D[0:3]**: 7-segment (3COM/3SEG) numeric digits with dots (hours:minutes, minutes:seconds in stopwatch mode or temperature)
3. Organized segments for animation:
   - **C0**: circular animation (for delay between repetitive alarm, 1/8 s in stopwatch mode …) driven by only one LCD data register: LCD_DATA15.
4. Standalone segments:
   - **B0/B1**: post/ante meridiem
   - **B2/B3**: degree Celsius/Fahrenheit

Table 5-1. Example of implementation.



	SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	…
COM0	E0	D0-b	D0-a	D0-f	D1-b	D1-a	D1-f	D2-b	…
COM1	E1	D0-c	D0-g	D0-e	D1-c	D1-g	D1-e	D2-c	…
COM2	E2	D0-dp	D0-d	/	D1-dp	D1-d	/	D2-dp	…
COM3	C0-a	C0-b	C0-c	C0-d	C0-e	C0-f	C0-g	C0-h	…

	…	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15	…
COM0	…	D2-a	D2-f	D3-b	D3-a	D3-f	A0-h	/	A0-a	…
COM1	…	D2-g	D2-e	D3-c	D3-g	D3-d	A0-i	A0-f	A0-b	…
COM2	…	D2-d	/	/	D3-d	/	A0-k	A0-e	A0-c	…
COM3	…	B0	B1	B2	B3	/	A0-n	A0-d	/	…

	…	SEG16	SEG17	SEG18	SEG19	SEG20	SEG21	SEG22	SEG23	…
COM0	…	A0-g	A1-h	/	A1-a	A1-g	A2h	/	A2a	…
COM1	…	A0-j	A1-i	A1-f	A1-b	A1-j	A2i	A2f	A2b	…
COM2	…	A0-l	A1-k	A1-e	A1-c	A1-l	A2k	A2e	A2c	…
COM3	…	A0-m	A1-n	A1-d	/	A1-m	A2n	A2d	/	…

	…	SEG24
COM0	…	A2g
COM1	…	A2j
COM2	…	A2l
COM3	…	A2m

NOTE C0-a and C0-b can individually blink.

6 Recommended reading

It is recommended to read the following documents to get an overall idea about Atmel AVR XMEGA B devices:

- XMEGA B Manual and Datasheets
- AVR1618: ATxmega B ASCII Character Mapping
- AVR1912: XMEGA-B1 Xplained Hardware User Guide
- AVR 1926: XMEGA-B1 Xplained Getting Started Guide

7 Table of contents

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Milennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan
16F, Shin Osaki Kangyo Bldg.
1-6-4 Osaki Shinagawa-ku
Tokyo 104-0032
JAPAN
Tel: (+81) 3-6417-0300
Fax: (+81) 3-6417-0370

8467A-AVR-11/12