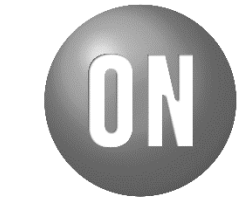
SOFTWARE MANUAL

LibAXDVK2(AXDVK2 Support Library)

Version 1.15

20180817



ON Semiconductor®

1. TABLE OF CONTENTS

2. INTRODUCTION

LibAXDVK2 is a convenience library to ease the use of AX-DVK2 Evaluation Board for the AX8052 Microprocessor. It contains the following features:

- Interrupt-driven Liquid Crystal Display access (libaxlcd2.h)
- Legacy polled Liquid Crystal Display access (libaxlcd.h)

LibAXDVK2 is available in source and binary form for SDCC, Keil C51 and IAR ICC.

3. Evaluation Board Peripherals

3.1. libaxlcd.h

libmflcd.h contains routines for accessing the alphanumeric 2x16 character liquid crystal display (LCD) on the evaluation board. libmflcd.h routines are blocking and do not require an interrupt. The routines may take a very long time to complete due to the slowness of the LCD.

3.1.1. void lcd_init(void)

This function initializes the interface and resets the LC display and sets it up.

3.1.2. void lcd_portinit(void)

This function only initializes the interface, but leaves the display alone. It can be used instead of lcd_init() when waking up from sleep or deepsleep while the display was kept powered, and it is undesirable if the display contents change.

3.1.3. void lcd_setpos(uint8_t v)

This function positions the LC display write cursor. Top line characters are numbered from left to right from 0x00 to 0x0F, and bottom line characters are numbered from 0x40 to 0x4F. The cursor does not auto-wrap from the top to the bottom line. After calling lcd_setpos(), lcd_waitshort() must be called or an equivalent delay must be kept before accessing the LCD again.

3.1.4. void lcd_writestr(const char *ch)

This function writes the null terminated C string pointed to by ch to the display at the current write cursor location. \n causes the output to continue at the beginning of the second line.

3.1.5. void lcd_writehex16(uint16_t val, uint8_t nrdig, uint8_t flags)
void lcd_writehex32(uint32_t val, uint8_t nrdig, uint8_t flags)

These functions write a hexadecimal number (val) to the display. The number of digits is given by nrdig. Leading characters are filled with zeros. flags is a bitwise or combination of the WRNUM constants documented in the LibMF manual.

3.1.6. void lcd_writenum16(uint16_t val, uint8_t nrdig, uint8_t flags)
void lcd_writenum32(uint32_t val, uint8_t nrdig, uint8_t flags)

These functions write a decimal number (val) to the display. The number of digits is given by nrdig. flags is a bitwise or combination of the WRNUM constants documented in the LibMF manual.

3.1.7. VOID LCD_CLEARDISPLAY(VOID)

This function clears the complete LC Display. It must be followed by lcd_waitlong, or the equivalent delay, before accessing the LC Display again.

3.1.8. VOID LCD_CLEAR(UINT8_T POS, UINT8_T LEN)

This function clears len characters starting at position pos.

3.1.9. VOID LCD_WAITLONG(VOID)
VOID LCD_WAITSHORT(VOID)

The LC display is very slow processing commands and data. These routines delay the microprocessor by a certain time by busy waiting. One of those routines, or the equivalent delay, should be called after each LC command or character. Which LC commands require the long delay is detailed in the LC display datasheet.

3.1.10. VOID LCD_WRITECMD(UINT8_T CMD)

This low level routine writes a command to the display.

3.1.11. VOID LCD_WRITEDATA(UINT8_T D)

This low level routine writes a character to the display.

3.2. LIBAXLCD2.H

libmflcd2.h contains routines for accessing the alphanumeric 2x16 character liquid crystal display (LCD) on the evaluation board. Unlike the routines from libmflcd.h, these routines are interrupt driven and (normally) nonblocking. They provide roughly the same interface as those from libmflcd.h, but libmflcd.h and libmflcd2.h routines must not be intermixed in the same project.

3.2.1. VOID LCD2_INIT(VOID)

This function initializes the interface and resets the LC display and sets it up.

3.2.2. VOID LCD2_PORTINIT(VOID)

This function only initializes the interface, but leaves the display alone. It can be used instead of lcd2_init() when waking up from sleep or deepsleep while the display was kept powered, and it is undesirable if the display contents change.

3.2.3. VOID LCD2_SETPOS(UINT8_T V)

This function positions the LC display write cursor. Top line characters are numbered from left to right from 0x00 to 0x0F, and bottom line characters are numbered from 0x40 to 0x4F. The cursor does not auto-wrap from the top to the bottom line.

3.2.4. VOID LCD2_WRITESTR(CONST CHAR *CH)

This function writes the null terminated C string pointed to by ch to the display at the current write cursor location. \n causes the output to continue at the beginning of the second line.

3.2.5. VOID LCD2_WRITEHEX16(UINT16_T VAL, UINT8_T NRDIG, UINT8_T FLAGS)
VOID LCD2_WRITEHEX32(UINT32_T VAL, UINT8_T NRDIG, UINT8_T FLAGS)

These functions write a hexadecimal number (val) to the display. The number of digits is given by nrdig. Leading characters are filled with zeros. flags is a bitwise or combination of the WRNUM constants documented in the LibMF manual.

3.2.6. VOID LCD2_WRITENUM16(UINT16_T VAL, UINT8_T NRDIG, UINT8_T FLAGS)
VOID LCD2_WRITENUM32(UINT32_T VAL, UINT8_T NRDIG, UINT8_T FLAGS)

These functions write a decimal number (val) to the display. The number of digits is given by nrdig. flags is a bitwise or combination of the WRNUM constants documented in the LibMF manual.

3.2.7. VOID LCD2_CLEARDISPLAY(VOID)

This function clears the complete LC Display.

3.2.8. VOID LCD2_CLEAR(UINT8_T POS, UINT8_T LEN)

This function clears len characters starting at position pos.

3.2.9. VOID LCD2_TX(UINT8_T V)

lcd2_tx writes one character to the LC display.

3.2.10. VOID LCD2_TXCMDSHORT(UINT8_T V)

lcd2_txcmdshort writes one command to the LC display. The command is expected to terminate within 50µs.

3.2.11. VOID LCD2_TXCMDLONG(UINT8_T V)

lcd2_txcmdlong writes one command to the LC display. The command is expected to terminate within 1.2ms.

3.2.12. UINT8_T LCD2_TXBUFFERSIZE(VOID)

This function returns the transmit buffer size. Buffer size is configurable.

3.2.13. UINT8_T LCD2_TXFREE(VOID)

lcd2_txfree returns the number of free space for characters in the FIFO.

3.2.14. UINT8_T LCD2_TXIDLE(VOID)

lcd2_txidle returns one if the LCD controller is idle, i.e. all transmit characters and commands have been sent to the LCD controller. This routine can be used to determine whether the microprocessor can enter sleep.

3.2.15. VOID LCD2_WAIT_TXDONE(VOID)

lcd2_wait_txdone blocks until the last character in the FIFO has been sent to the LCD controller.

3.2.16. VOID LCD2_WAIT_TXFREE(UINT8_T V)

lcd2_wait_txfree blocks until the number of free space for characters in the FIFO reaches or exceeds v.

3.2.17. VOID LCD2_TXADVANCE(UINT8_T IDX)

lcd2_txadvance adds the given number of characters at the end of the transmit FIFO. They must have been defined by lcd2_txpoke or lcd2_txpokehex before, otherwise the transmitted characters are undefined.

3.2.18. VOID LCD2_TXPOKE(UINT8_T IDX, UINT8_T CH)

lcd2_txpoke puts a character at the idx'th position after the end of the FIFO. It does not become part of the FIFO. In order to actually transmit the character, lcd2_txadvance must be called.

3.2.19. VOID LCD2_TXPOKEHEX(UINT8_T IDX, UINT8_T CH)

lcd2_txpoke puts a hexadecimal character at the idx'th position after the end of the FIFO. It does not become part of the FIFO. In order to actually transmit the character, lcd2_txadvance must be called.

3.2.20. UINT8_T LCD2_POLL(VOID)

Normally, data is transferred between the FIFO and the hardware using an interrupt handler. Sometimes though, an interrupt handler is undesirable; in this case, lcd2_poll can be called periodically to transfer data between the FIFO and the hardware if available. It returns a bit mask with bit 1 set if a byte was transferred from the FIFO to the hardware.

3.2.21. LCD2_DEFINE_TXBUFFER(SZ)

This macro defines the transmit buffer size. The argument needs to lie between 14 and 256 (inclusive). Note that the argument specifies the total buffer size; one command is unusable due to the design of the buffer pointers, therefore lcd2_txbuffersize will return one command less than the argument. The number of xdata bytes allocated is twice the argument.

3.3. LIBDVK2LEDS.H
LIBMINIKITLEDS.H

These convenience headers provide macros to manipulate the LED's of the DVK2 and the Minikit boards, respectively.

3.3.1. LED0_STATE(X)
LED1_STATE(X)
LED2_STATE(X)
LED3_STATE(X)

These macros set the state of the corresponding LED. Zero means off, one means on.

3.3.2. LED0_ON()
LED1_ON()
LED2_ON()
LED3_ON()

These macros turn the corresponding LED on.

3.3.3. LED0_OFF()
LED1_OFF()
LED2_OFF()
LED3_OFF()

These macros turn the corresponding LED off.

3.3.4. LED0_TOGGLE()
LED1_TOGGLE()
LED2_TOGGLE()
LED3_TOGGLE()

These macros toggle the corresponding LED on or off.

4. Contact Information

ON Semiconductor
Oskar-Bider-Strasse 1
CH-8600 Dübendorf
SWITZERLAND

Phone +41 44 882 17 07
Fax +41 44 882 17 09
Email sales@onsemi.com
www.onsemi.com

For further product related or sales information please visit our website or contact your local representative.