

## USING THE LCD ON KWIKBYTE KB9202

## 1 Introduction

This document shows LCD usage on the KwikByte KB9202 development board.

The terms ‘KB9202’ and ‘KB9202B’ are used to refer to the KB9202 development board and should be considered equivalent.

### DISCLAIMER:

The information provided here is for reference only. No warranty of ANY kind is provided. KwikByte assumes no liability for the use of this information in any application. All trademarks, patents, and other rights remain with the respective owner(s).

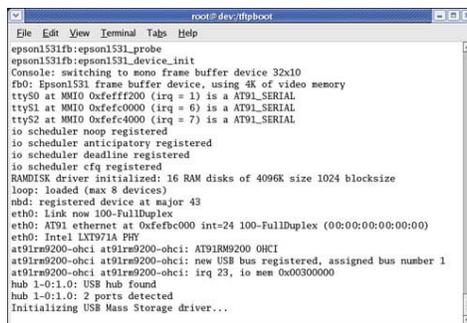
### 1.1 Linux Kernel Configuration

The kernel configuration file (arch/arm/configs/kb9202\_defconfig) has been updated to include the LCD driver, by default. The factory installation image has been updated to include the new kernel (2.6.20 at time of this writing).

The default kernel configuration selects the 4x6 mini font as the sole, built-in font. Another font may be selected according to user preference.

### 1.2 Boot Loader Configuration

By default, the boot loader passes arguments to the kernel specifying console support on device ttyS0. To view the kernel boot progress on the LCD, remove “console=ttyS0,115200” from the boot arguments.

A screenshot of a terminal window titled 'root@dev:/#ipboot'. The terminal displays the following text:

```
File Edit View Terminal Tags Help
epson1531fb:epson1531_probe
epson1531fb:epson1531_device_init
Console: switching to mono frame buffer device 32x10
fb0: Epson1531 frame buffer device, using 4K of video memory
ttyS0 at MMIO 0xfefc200 (irq = 1) is a AT91_SERIAL
ttyS1 at MMIO 0xfefc000 (irq = 6) is a AT91_SERIAL
ttyS2 at MMIO 0xfefc400 (irq = 7) is a AT91_SERIAL
io scheduler noop registered
io scheduler anticipatory registered
io scheduler deadline registered
io scheduler cfq registered
RAMDISK driver initialized: 16 RAM disks of 4096K size 1024 blocksize
loop: loaded (max 8 devices)
nbd: registered device at major 43
eth0: Link now 100-FullDuplex
eth0: AT91 ethernet at 0xfefbc00 int=24 100-FullDuplex (00:00:00:00:00:00)
eth0: Intel LXT971A PHY
at91rm9200-ohci at91rm9200-ohci: AT91RM9200 OHCI
at91rm9200-ohci at91rm9200-ohci: new USB bus registered, assigned bus number 1
at91rm9200-ohci at91rm9200-ohci: irq 23, io mem 0x00300000
hub 1-0:1.0: USB hub found
hub 1-0:1.0: 2 ports detected
Initializing USB Mass Storage driver...
```

Figure 1: Sample terminal boot screen

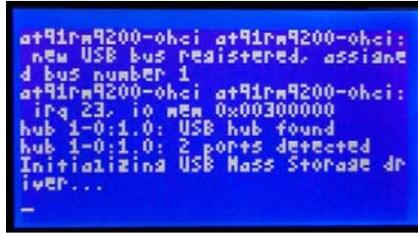


Figure 2: Sample LCD boot screen (4x6 minifont)

## 1.3 Applications

### 1.3.1 Text Output from Command Line

The display can be written directly from the command line using redirection. For example, the command

```
echo "Output text to /dev/tty0" > /dev/tty0
```

produces the following output:

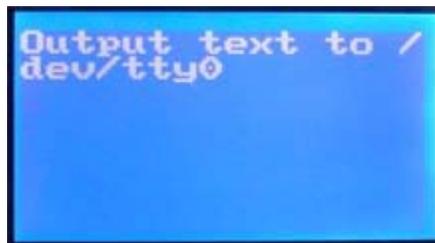


Figure 3: Command line output (8x8 font)

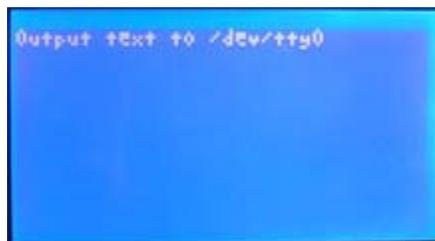


Figure 4: Command line output (4x6 minifont)

### 1.3.2 Text Output from Program

Programs can also write to the display device using standard file I/O operations:

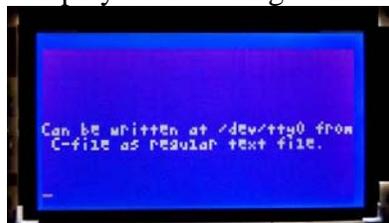


Figure 5: File I/O output (4x6 minifont)

### 1.3.3 Graphical Display

Because the LCD driver is implemented as a frame buffer device, graphical image can be displayed on the device using standard file I/O operations as well. A sample application is included which interfaces with a common USB mouse and shows moving bar graphs while displaying the current system time.



Figure 6: Sample graphics application

This threaded application ends when the user presses the 'Enter' key or the mouse is clicked on the 'X' quit box (top right corner of the image).

Source code for the sample program is provided. The application can be easily adapted to fit numerous practical applications.

## 1.4 Notes

### 1.4.1 Blanking

The screen is 'blanked' in the following conditions:

- 1) System specified time-out with inactivity
- 2) Writing to the corresponding sysfs device:  
`echo "1" > /sys/class/graphics/fb0/blank`

The screen is 'unblanked' in the following conditions:

- 1) Activity on the display: e.g., move the USB mouse
- 2) Writing to the corresponding sysfs device:  
`echo "0" > /sys/class/graphics/fb0/blank`

If you want to disable the blanking operation entirely, execute the following command:

```
/bin/no_blank
```

This writes a binary sequence to the device (take a look at the script).

### 1.4.2 Backlight

The backlight can be controller by blanking operations, described above, or manually.

Turn on the backlight:

```
echo "0" > /sys/class/backlight/kb9202-bl/power
```

Turn off the backlight:

```
echo "1" > /sys/class/backlight/kb9202-bl/power
```