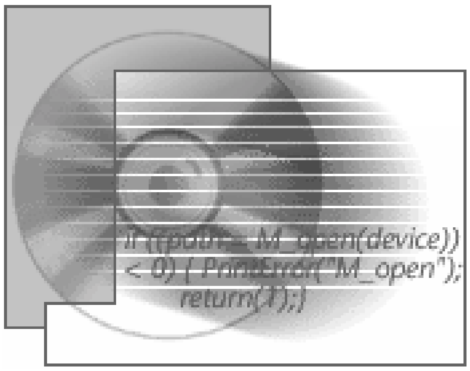
Embedded Solutions

Using P1/P501 Graphics on MEN 824x/ALI boards under ELinOS



Application Note

Board-Level Computers for Industrial Applications

mikro elektronik gmbh · nürnberg

About this Document

This document describes how to configure P1/P501 graphics mezzanines under ELinOS 2.1 on MEN’s 824x/ALI series PowerPC boards (F1N, B11, A12, D3, SC13).

When you have followed these instructions, you'll be able to:

- use a VGA monitor as a Linux text console.
- access the graphics memory from an application program through */dev/fb0*.
- run QT/embedded programs.
- run X-Window system server (on a frame buffer basis).

History

Edition	Description	Technical Content	Date of Issue
E1	First edition	Klaus Popp	2002-12-18

Conventions

⚠ This sign marks important notes or warnings concerning proper functionality of the product described in this document. You should read them in any case.

italics Folder, file and function names are printed in *italics*.

bold **Bold** type is used for emphasis.

`monospace` A `monospaced` font type is used for listings, C function descriptions or wherever appropriate.

hyperlink Hyperlinks are printed in blue color.

The globe will show you where hyperlinks lead directly to the Internet, so you can look for the latest information online.

Copyright Information

Please recycle

Germany
MEN Mikro Elektronik GmbH
Neuwieder Straße 7
90411 Nuremberg
Phone +49-911-99 33 5-0
Fax +49-911-99 33 5-99
E-mail info@men.de
www.men.de

France
MEN Mikro Elektronik SA
18, rue René Cassin
ZA de la Châtelaine
74240 Gaillard
Phone +33 (0) 450-955-312
Fax +33 (0) 450-955-211
E-mail info@men-france.fr
www.men-france.fr

UK
MEN Micro Ltd
Whitehall, 75 School Lane
Hartford, Northwich
Cheshire UK, CW8 1PF
Phone +44 (0) 1477-549-185
Fax +44 (0) 1477-549-178
E-mail info@menmicro.co.uk
www.menmicro.co.uk

USA
MEN Micro, Inc.
3740 North Josey Lane, Suite 203
Carrollton, TX 75007
Phone 972-939-2675
Fax 972-939-0055
E-mail sales@menmicro.com
www.menmicro.com

Contents

1 Getting Started

1.1 Host Requirements

- Linux PC (SuSE, RedHat...)
- ELinOS 2.1/PowerPC development kit installed on your development Linux PC.

1.2 Target Prequisites

You must have a P1/P501 graphics module with the silicon motion chip SMI 710 or 712 (SMI 910 should work as well, but has not been tested).

☑ Mount the graphics module on any PC•MIP/PMC slot.

☑ Attach a Multi-Sync monitor to the VGA connector of the graphics module.

☑ Attach PS/2 keyboard/mouse to the target (see hardware manual of your board).

☑ You need to have a least the following MENMON (Firmware) versions on your PowerPC board: (Older MENMON versions always initialize the graphics card in text mode.)

- F1N: MENMON 5.0
- B11: MENMON 3.0
- A12/D3/SC13: MENMON 3.0

Contact support@men.de to receive a MENMON update for your board.

2 Configuring Video Mode

The video mode to be used can be configured only through MENMON. The Linux frame buffer driver just takes over the settings made by MENMON. By default, MENMON comes up in VGA text mode (mode 3).

In order to use the Linux frame buffer device, video mode must be changed to one of the following graphic modes:

Table 1. *Supported Video Modes*

Video Mode Number (hex)	Description
3	640 x 480, VGA text mode
101	640 x 480, 8-bit indexed colors
111	640 x 480, 16-bit 5-6-5 RGB colors
112	640 x 480, 32-bit x-8-8-8 RGB colors
103	800 x 600, 8-bit indexed colors
114	800 x 600, 16-bit 5-6-5 RGB colors
115	800 x 600, 32-bit x-8-8-8 RGB colors
105	1024 x 768, 8-bit indexed colors
117	1024 x 768, 16-bit 5-6-5 RGB colors
118	1024 x 768, 32-bit x-8-8-8 RGB colors
107	1280 x 1024, 8-bit indexed colors
11A	1280 x 1024, 16-bit 5-6-5 RGB colors

All video modes operate at a refresh rate of 75 Hz.

Note: For the beginning, you should choose a mode that uses "8-bit indexed colors". See Chapter 5 Using Qt/Embedded on page 10.

For example, to change the video mode to "800 x 600, 8-bit indexed colors", enter

```
MenMon> ee-vmode 103
```

The video mode is then immediately changed, and the setting is stored in the EEPROM (will not be lost when system is powered off).

Warning: If the resulting video frequency exceeds the limits of your monitor, your monitor might be damaged!

3 Enabling Frame Buffer Support in ELinOS

3.1 Preparing the Kernel Sources

When this document was created (see Chapter History on page 2), the ELinOS distribution did not support P1/P501 graphics mode for MEN's 824x ALI boards. You need to make some small changes to the Linux kernel:

File */opt/elinos/linux-2.4.18/arch/ppc/platforms/men824xali_setup.c*

Insert the following lines at the end of function *men824xali_setup_arch()*:

```
      ...
#ifdef CONFIG_DUMMY_CONSOLE
      conswitchp = &dummy_con;
#endif
}
```

File */opt/elinos/linux-2.4.18/drivers/video/smiLynx.c*

Change the following section in function *smi_setcolreg()*:

Old, buggy version:

```
#ifdef FBCON_HAS_CFB8
      case 8:
          outb( 0xFF, SMI_LUT_MASK );
          outb( (unsigned char) regno, SMI_LUT_START );

          outb( (red & 0xff)   >> 2, SMI_LUT_RGB );
          udelay(10);
          outb( (green & 0xff) >> 2, SMI_LUT_RGB );
          udelay(10);
          outb( (blue & 0xff)  >> 2, SMI_LUT_RGB );
          break;
#endif
```

Corrected version:

```
ifdef FBCON_HAS_CFB8
      case 8:
          outb( 0xFF, SMI_LUT_MASK );
          outb( (unsigned char) regno, SMI_LUT_START );
         /*
          * popp@men.de: when using QT embedded, only the upper byte
          * is valid. When called from fb_con, upper and lower byte of
          * red, green, blue contain same value
          */
          outb( (red & 0xff00)   >> 10, SMI_LUT_RGB );
          udelay(10);
          outb( (green & 0xff00) >> 10, SMI_LUT_RGB );
          udelay(10);
          outb( (blue & 0xff00)  >> 10, SMI_LUT_RGB );
          break;
#endif
```

3.2 Configuring the Kernel for Frame Buffer Support

You can now clone any ELinOS demo, as usual (using *elinos-cloneproject*). To enable frame buffer support, make the following changes. Go into your ELinOS project directory and call *elk* > *Kernel Configuration*:

"Console drivers ->

"Support for VGA Console" (*CONFIG_VGA_CONSOLE*) **no**

"Console drivers -> Frame-buffer support"

"Support for frame buffer devices" (*CONFIG_FB*) **yes**

"SMI 710 (LynxEM) display support" (*CONFIG_FB_SMI710*)**yes**

"Advanced low level driver options" (*CONFIG_FBCON_ADVANCED*) **yes**

"8 bpp packed pixels support" (*CONFIG_FBCON_CFB8*) **yes**

"16 bpp packed pixels support" (*CONFIG_FBCON_CFB16*) **yes**

"32 bpp packed pixels support" (*CONFIG_FBCON_CFB32*) **yes**

"Select compiled-in fonts" (*CONFIG_FBCON_FONTS*) **yes**

"VGA 8x16 font" (*CONFIG_FONT_8x16*) **yes**

"Character devices -> Mice"

"Mouse Support (not serial and bus mice)" (*CONFIG_MOUSE*) **yes**

"PS/2 mouse (aka "auxiliary device") support" (*CONFIG_PSMOUSE*) **yes**

Now you should recompile the kernel and build your boot files.

4 Using the Frame Buffer Console

The frame buffer console (i.e. virtual text mode console) can be used with any resolution and color depth that can be set up by MENMON.

If you have connected a PS/2 keyboard and an SMI graphics chip was found, the Linux console is automatically redirected to your VGA console. The boot messages will also appear there.

If you don't have connected a keyboard, then MENMON tells the Linux kernel to use the serial console (through MENMON command line parameter *cons*=). In this case, you can still access the VGA text mode console under */dev/vc/0* (if you're using the device file system) or */dev/vcs0*.

5 Using Qt/Embedded

The *smiLynx* frame buffer driver for the P1/P501 can also be used to run Qt/Embedded applications.

However, this driver will currently not support the SMI chip's drawing engine, therefore all drawing actions are done in software (i.e. no hardware acceleration).

Currently, Qt only works correctly for modes using 8-bit, indexed colors. In all other modes, the colors are not displayed correctly. This problem occurs because the SMI chip is connected to the little-endian PCI bus, but the PowerPC is a big-endian system and the bytes within 16/32-bit values are swapped.

Table 2. *Supported Video Modes for Qt/Embedded*

Video Mode Number (hex)	Description
101	640 x 480, 8-bit indexed colors
103	800 x 600, 8-bit indexed colors
105	1024 x 768, 8-bit indexed colors
107	1280 x 1024, 8-bit indexed colors

5.1 Kernel Configuration for QTE Demo

You already must have prepared the kernel sources as described in Chapter 3.1 Preparing the Kernel Sources on page 7.

For your first experiments, you should try to run the QTE demo shipped with ELinOS. However, when this document was created (see Chapter History on page 2), the MEN 824x ALI platforms were not supported in this demo, therefore we must do some steps manually.

```
kp@kplinux:~> /opt/elinos/bin/elinos-cloneproject \
/opt/elinos/demos/QTE elinosQTEP1

CLONING PROJECT `elinosQTEP1' FROM `/opt/elinos/demos/QTE'
==========================================================

Checking existing project /opt/elinos/demos/QTE... ok
Checking new project elinosQTEP1... ok
Cloning project /opt/elinos/demos/QTE as elinosQTEP1... ok

Kernel Source Tree [linux-2.4.18-kp]:
Checking Kernel-Source /opt/elinos/linux-2.4.18-kp... ok
Setting up new kernel... ok

CONFIGURING PROJECT
===================

Configuration: reading from ./project.config... ok
```

```
Current Settings:

   ELINOS_BOARD      = custom
   ELINOS_CPU        = ppc
   ELINOS_ARCH       = 4xx
   ELINOS_LIBC       = libc6
   ELINOS_DOSNAME    = qte
   ELINOS_BOOT_STRAT = floppy

Board Type [custom]:
CPU Type [ppc]:
CPU Architecture [4xx]: 60x
Version of libc to work with [libc6]:
Project Name (8 characters at most, no blanks) [qte]:
You must now select the boot strategy for your new project:
Boot Strategy: [floppy]: ppcboot_multi
  Boot Image Load Address [00000000]:
  Boot Image Entry Address [00000000]:
Change kernel configuration to defconfig [y]:
Kernelconfiguration: using linux/arch/ppc/defconfig

Writing file ./project.config... ok
Writing file ELINOS.sh... ok
Configuring /home/kp/elinosQTEP1/linux... ok

---------------------------------------------
Your new project has been set up successfully.
To work on it, type:

  sh# cd "/home/kp/elinosQTEP1"
  sh# . ELINOS.sh
---------------------------------------------

CLONING DONE.
```

You should then copy a default configuration for your CPU board used, for example an A12 board:

```
kp@kplinux:~> cp /opt/elinos/demos/BusyBox/kernelconfigs/mena12_config
elinosQTEP1/linux/.config
```

Then start *elk* and add frame buffer support as described in Chapter 3.2 Configuring the Kernel for Frame Buffer Support on page 8.

In addition, you must make the following changes to the kernel configuration:

"File Systems" ->

"/dev file system support" (*CONFIG_DEVFS_FS*) **no**

"General Setup" ->

"Networking support" (*CONFIG_NET*) **yes**

"Networking Options" ->

"Unix domain sockets" (*CONFIG_UNIX*) **yes**

"TCP/IP networking" (CONFIG_INET) yes

You also need to change the mouse device that is used by the QTE demo program.

Change *elinosQTEP1/src/hello/main.src*:

Make sure that Mouse is set to "Auto:/dev/psaux" e.g.:

```
/* ppc setenv( "QWS_MOUSE_PROTO", "Auto:/dev/psaux", 1); */
```

Note: *main.src* may change between different releases of ELinOS.

6 Accessing the Frame Buffer Memory from Your Own Program

The following small example shows how to open the frame buffer device and draw some rectangles on the screen:

(This example works only for 8 bits/color, but further modes can easily be implemented.)

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <errno.h>
#include <ctype.h>
#include <string.h>
#include <linux/fb.h>
#include <sys/mman.h>
#include <sys/ioctl.h>

#define FBDEV "/dev/fb0"

struct fb_var_screeninfo G_var;
struct fb_fix_screeninfo G_fix;

void perror_msg_and_die(char *str)
{
      perror(str);
      exit(1);
}

void dumpBitField( struct fb_bitfield *bf )
{
      printf( "off=%d len=%d msb_right=%d\n",
            bf->offset, bf->length, bf->msb_right);
}

void dumpVar( struct fb_var_screeninfo *var )
{
      printf("fb_var_screeninfo:\n");
      printf(" res=%dx%d\n", var->xres, var->yres );
      printf(" bits_per_pixel=%d\n",  var->bits_per_pixel );
      printf(" Color bitfields:\n");
      printf("  Red  : " ); dumpBitField( &var->red );
      printf("  Green: " ); dumpBitField( &var->green );
      printf("  Blue : " ); dumpBitField( &var->blue );
      printf("  Trans: " ); dumpBitField( &var->transp );
      printf(" Nonstd pixel fmt=%d\n", var->nonstd );

}

void dumpFix( struct fb_fix_screeninfo *fix )
{
      printf("fb_fix_screeninfo: id=%s\n", fix->id);
      printf(" phys fb start=0x%08lx, size=0x%08x\n", fix->smem_start,
            fix->smem_len );
      printf(" fb type (FB_TYPE_xx)=%d \n", fix->type );
      printf(" visual (FB_VISUAL_xx)=%d (2=truecol 3=pseudocol 4=directcol "
            "5=staticpseudo)\n", fix->visual );
      printf(" line len=%d\n", fix->line_length );
      printf(" accel=%d\n", fix->accel );
}

void pseudoTestPic( int fh, void *mapFb )
{
      __u16 red[] = { 0xffff, 0x0000, 0x0000, 0xffff, 0x2000 };
      __u16 green[] = { 0x0000, 0xffff, 0x0000, 0xffff, 0x2000 };
      __u16 blue[] = { 0x0000, 0x0000, 0xffff, 0xffff, 0x2000 };
      enum { _red=0, _green=1, _blue=2, _white=3, _grey=4, _numcols=5 };
      struct fb_cmap cmap = {
         0,
         _numcols,
         red, green, blue, NULL
      };
      int bs = G_fix.line_length*120;

      if (ioctl(fh, FBIOPUTCMAP, &cmap))
         perror_msg_and_die("fbset(FBIOPUTCMAP)");

      memset( (char *)mapFb+(bs*0), _red, bs );
      memset( (char *)mapFb+(bs*1), _blue, bs );
      memset( (char *)mapFb+(bs*2), _green, bs );
      memset( (char *)mapFb+(bs*3), _white, bs );


}


int main( int argc, char **argv )
{
      int fh;
      void *mapFb;

      printf("FBTEST\n");
      if ((fh = open(FBDEV, O_RDWR)) < 0)
         perror_msg_and_die("open " FBDEV);
      if (ioctl(fh, FBIOGET_FSCREENINFO, &G_fix))
         perror_msg_and_die("fbset(FBIOGET_FSCREENINFO)");
      if (ioctl(fh, FBIOGET_VSCREENINFO, &G_var))
         perror_msg_and_die("fbset(FBIOGET_VSCREENINFO)");

      dumpFix(&G_fix);
      dumpVar(&G_var);

      mapFb = mmap( NULL, G_fix.smem_len,
                 PROT_READ|PROT_WRITE,
                 MAP_SHARED, fh, 0 );
      if( mapFb == MAP_FAILED )
         perror_msg_and_die("mmap");
      if( G_fix.visual == 3 )
         pseudoTestPic( fh, mapFb );

      return 0;
}
```

7 Using X Window Server

On top of the frame buffer device */dev/fb0*, you can use the X-server *XF86_FBDev* which is part of the ELinOS distribution.

The X-Server works with the same restrictions as *Qt/embedded*; i.e. you can use only modes with 8 bits/color.

For more information on how to set up the X-Server, see the "man" pages of *XF86_FBDev* and *XFree86*.

Unfortunately, there is no ready-to-run demo project for ELinOS that includes an X-Server. If you have questions, please contact support@sysgo.de.

Fax Reply

mikro elektronik
gmbh • nürnberg

Who you are...

Name	____	**Phone No.**	____
Company	____	**Fax No.**	____
Department	____	**E-mail**	____

Use our online forms at www.men.de

- **Technical Support**
- **User Manuals**

What we can do for you...

Product Support

Product Article No. ____

Revision __ __ . __ __ . __ __

If the product is a mezzanine module:

Carrier Article No. ____

Revision __ __ . __ __ . __ __

Operating system

❍ OS-9

❍ VxWorks

❍ WindowsNT

❍ other:

____

Manual Feedback

Manual Article No. ____

Edition E ___

Useful or awful?

very useful ❍ ❍ ❍ ❍ ❍ totally awful

Usage

❑ I read the manual and decided to buy the related product.

❑ I read the manual when I got the product.

❑ I refer to the manual only when I have problems.

Your problems/comments...

MEN Mikro Elektronik GmbH

Neuwieder Straße 7
90411 Nürnberg
Deutschland

Tel. +49-911-99 33 5-0
Fax +49-911-99 33 5-99

E-Mail info@men.de
www.men.de