MiniGUI User Manual

Version 2.0 (revised edition 4)

For MiniGUI Version 2.0.4/1.6.10

Beijing Feynman Software Technology Co. Ltd.

August, 2007

Copyright Claim

MiniGUI User Manual Version 2.0 (revised edition 4) for MiniGUI Ver 2.0.4/1.6.10.

By whatever means you get the entire or partial text or photograph data in this guide, no matter mechanical or electronic, you are only authorized by Beijing Feynman Software Technology Co., Ltd. the reading right. Any format conversion, redistribution, dissemination, and copying its entire or partial content, or using text or photograph therein for commercial purpose without written permission will be regarded as tortuous, and may result in severe civil or criminal punishment.

Contents

1 Introduction to MiniGUI Value-added Release

1.1 A Brief Introduction

MiniGUI, developed by Beijing Feynman Software Technology Co. Ltd., originates from a world famous free software project, which is initiated by Wei Yongming. MiniGUI aims to provide a fast, stable and lightweight graphics user interface (GUI) support system for real-time embedded systems. MiniGUI is "a cross-operating-system graphics user interface support system for embedded devices", and "an embedded graphics middleware". After over nine years of development since the end of 1998, MiniGUI has become a stable and reliable one for widespread application in a variety of products and programs; it can run on Linux/uClinux, eCos, VxWorks, pSOS, ThreadX, Nucleus, OSE, and even uC/OS-II, also on the Win32 platform.

MiniGUI defines a set of lightweight windowing and graphics interfaces for applications. Using these interfaces, an application can create multiple main windows and controls in them, such as buttons and edit boxes. MiniGUI provides powerful graphics functions for developers, helping to display all kinds of bitmaps and draw complicated graphics in windows.

Feynman Software provides MiniGUI Value-Added Release (MiniGUI-VAR) product for commercial users who buy MiniGUI commercial licenses. Besides this, Feynman Software releases two GPL versions of MiniGUI: MiniGUI V1.3.3 and MiniGUI-STR V1.6.2. The GPL versions, including its source code, development documentations and demo programs, are available at the following web page for free download:

```
http://www.minigui.com/download/cmgother.shtml
```

However, the versions that you download freely from our site would be only used to develop GPL applications. If you are using MiniGUI for developing commercial applications or other software that are not covered by the terms listed in GPL, you should have a commercial license for MiniGUI from Feynman Software.

Currently, MiniGUI Value-Added Release (MiniGUI-VAR) can be divided into two versions: MiniGUI V2.0.x provides support for multi-process-based operating systems, like Linux; MiniGUI v1.6.x provides support for traditional real-time embedded operating systems, which are multi-thread- or multi-task- based. The former provides support for the runtime modes MiniGUI-Processes and MiniGUI-Threads, and the later provides support for the runtime mode MiniGUI-Threads.

MiniGUI-VAR is divided into a series of products according to the operating systems, please see Table 1.1. Table 1.1 also illustrates the runtime mode(s) provided by the products.

Table 1.1 MiniGUI-VAR products and runtime modes supported

Products and versions	Runtime mode(s) supported
MiniGUI-VAR for Linux V2.0.x	MiniGUI-Processes MiniGUI-Threads MiniGUI-Standalone
MiniGUI-VAR for uClinux V1.6.x	MiniGUI-Threads MiniGUI-Standalone
MiniGUI-VAR for VxWorks V1.6.x	MiniGUI-Threads
MiniGUI-VAR for ThreadX V1.6.x	MiniGUI-Threads
MiniGUI-VAR for Nucleus V1.6.x	MiniGUI-Threads
MiniGUI-VAR for OSE V1.6.x	MiniGUI-Threads
MiniGUI-VAR for pSOS V1.6.x	MiniGUI-Threads
MiniGUI-VAR for eCos V1.6.x	MiniGUI-Threads
MiniGUI-VAR for uC/OS-II V1.6.x	MiniGUI-Threads

Except for the difference of runtime modes supported, these two versions have the almost same features.

For the detailed description about runtime modes and MiniGUI features, please refer to *MiniGUI Technology White paper for V2.0.4/1.6.10* and *Datasheet for MiniGUI V2.0.4/1.6.10*.

1.2 Documents for MiniGUI-VAR

Except for this manual, Feynman Software also ships the following printed handbook with the MiniGUI-VAR product:

- *MiniGUI Programming Guide* Version 2.0-4. This guide describes in detail the foundation knowledge of MiniGUI on developing embedded application software, technical documents and development skills, the content of which involves various aspects of MiniGUI programming, include message looping, window procedure, dialog box, controls, graphics interfaces, and so on.

In the directory **`minigui/docs/`** of MiniGUI-VAR CD-ROM, you can find the document files for this manual and *MiniGUI Programming Guide* Version 2.0-4 in PDF format. Besides these files, there are the following documents (in PDF format) in the above directory:

- *MiniGUI API Reference Manual* for MiniGUI Version 2.0.4. This manual describes the APIs of MiniGUI V2.0.4 (MiniGUI-Processes runtime mode) in detail[1].
- *MiniGUI API Reference Manual* for MiniGUI Version 1.6.10. This manual describes the APIs of MiniGUI V1.6.10 (MiniGUI-Threads runtime mode) in detail[2].
- *MiniGUI Technology White paper for V2.0.4/1.6.10* and *Datasheet for MiniGUI V2.0.4/1.6.10.*

README file located in the product CD-ROM root directory describes the file name and the location of above documents. There is also ReleaseNotes.pdf file in this directory. This file describes the new features, enhancements, and optimizations in this release. Please pay special attention to the backward compatibility issues.

1.3 MiniGUI Source Code and Samples

In the MiniGUI-VAR product CD-ROM, there are the source code packages of MiniGUI, which is specific to the version and the operating system.

For MiniGUI-VAR V2.0.4, the source packages are located in the directory **`minigui/2.0.x`** of the product CD-ROM. The packages are listed as follow:

- **`libminigui-2.0.4-<os>.tar.gz`**: The source package of MiniGUI V2.0.4 for `<os>` (like Linux) operating system. MiniGUI is composed of three libraries: libminigui (source is in **`src/`**), libmgext (**`ext/`**), and libvcongui (**`vcongui/`**). Libminigui is the core library, which provides window management support and

[1] Only English edition in HTML format and Windows CHM format
[2] Only English edition in HTML format and Windows CHM format

graphics interfaces as well as standard controls. Libmgext is an extension library of libminigui; it provides some useful controls and convenient functions, such as `Open File Dialog Box'. Libvcongui provides a virtual console window in which you can run programs. Libmgext and libvcongui have already been contained in this package.

- **`minigui-res-2.0.4.tar.gz`**: Runtime resources required by MiniGUI including fonts, icons, bitmaps, and cursors.
- **`mg-samples-2.0.4.tar.gz`**: The sample program package for *MiniGUI Programming Guide*.
- **`mde-2.0.4.tar.gz`**: The MiniGUI demo program package, which provides some complex demo applications, such as notebook, housekeeper, and minesweeper.

For MiniGUI-VAR V1.6.10, the source packages are located in the directory **`minigui/1.6.x`** of the product CD-ROM. The packages are listed as the follow:

- **`libminigui-1.6.10-<os>.tar.gz`**: The source package of MiniGUI V2.0.4 for `<os>` (like vxworks) operating system. MiniGUI is composed of three libraries: libminigui (source is in **`src/`**), libmgext (**`ext/`**), and libvcongui (**`vcongui/`**). Libminigui is the core library, which provides window management support and graphics interfaces as well as standard controls. Libmgext is an extension library of libminigui; it provides some useful controls and convenient functions, such as `Open File Dialog Box'. Libvcongui provides a virtual console window in which you can run programs. Libmgext and libvcongui have already been contained in this package.
- **`minigui-res-1.6.10.tar.gz`**: Runtime resources required by MiniGUI including fonts, icons, bitmaps, and cursors.
- **`mg-samples-1.6.10.tar.gz`**: The sample program package for *MiniGUI Programming Guide*.
- **`mde-1.6.10.tar.gz`**: The MiniGUI demo program package, which provides some complex demo applications, such as notebook, housekeeper, and minesweeper.

[NOTE] Although Feynman Software provides the complete MiniGUI source code for you, you are only permitted to add new graphics engine and input engine in order to support different hardware; you are not permitted to modify other source code of MiniGUI.

1.4 Other Resources in the Product CD-ROM

Except for the source code packages and documents, there are other resources in the product CD-ROM:

- **`deplibs/`**: There are the source code packages of **`freetype`**, **`libjpeg`**, **`libpng`**, **`zlib`**, and so on.
- **`win32-dev/`**: There is the MiniGUI development package for Win32 platform.
- In other directories, there may be some development tools and software for specific operating systems.

1.5 Optional Components of MiniGUI-VAR

Except for the MiniGUI-VAR product, Feynman Software also provides some MiniGUI component products and other MiniGUI applications such as mSpider. Figure 1.1 shows the product line of Feynman Software.

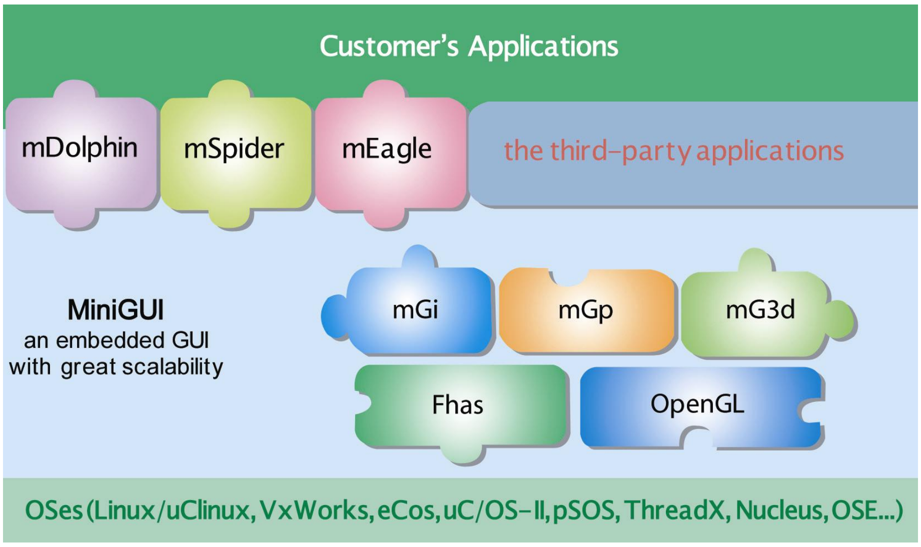


Figure 1.1 Product line of Feynman Software

mGi provides input method framework for applications based on MiniGUI. mGi now provides the framework for soft-keyboard and hand writing input methods. mGi also provides an IME container for user to add self-defined IME to it. On the other hand, you can use self-defined keyboard bitmap for the soft-keyboard and add your self-defined translation method to it.

mGp provides a printing engine for applications based on MiniGUI so that applications using mGp will have the printing function. At present, mGp provides printing support for Epson, HP and some other printers. Note that mGp only provides the support for Linux operating system.

mG3d is a 3D rendering library for applications based on MiniGUI. By using this library, you can render 3D objects in your applications.

Except for these three component products above, Feynman Software also provides MiniGUI SDK for Win32. By using MiniGUI SDK for Win32, you can run MiniGUI and its applications on Win32 platform. You can even write and debug MiniGUI applications by using Visual Studio IDE tool. However, there are some limitations:

- MiniGUI SDK for Win32 only provides the support for the runtime MiniGUI-Threads.
- When you use MiniGUI SDK for Win32 to develop MiniGUI application, please do not invoke any function specific to Win32, because the function may not exist on your target operating system.

For the complete Feynman products, please visit the following web page:

```
http://www.minigui.com/product/cindex.shtml
```

1.6 About this Manual

This manual mainly describes the compile-time configuration options and the runtime configuration options of MiniGUI.

2 Configuring, Compiling, and Installing MiniGUI

In general, Embedded Systems are special systems, and they have different requirement for graphics system. Some system required a basic graphics function but some one required a complete graphics, window and controls supporting. So an embedded graphics system must be constituted. MiniGUI provides a lot of configuration options. You can specify the functions of MiniGUI library. Generally, we can configure MiniGUI as follows:

- Specify the operating system and the target board on which MiniGUI runs.
- Specify MiniGUI running mode: MiniGUI-Threads base on thread, MiniGUI-Processes based on processes or the simple MiniGUI-Standalone.
- Specify the graphics engine and the input engine, as well as the options of these engines.
- Specify font class supported and the type of incore fonts.
- Specify the supporting character set.
- Specify the supporting image file format.
- Specify the supporting control class.
- Specify the style of the controls, i.e. CLASS style, FLAT style or FASHION style.

In this chapter we will discuss the compiling configuration options, in order that user can create a most suitable MiniGUI for their embedded system. We will discuss the compiling and installing of MiniGUI too.

2.1 Customization of Compiling Configuration Options

A file named **`mgconfig.h`** is located in the root directory of MiniGUI source code. A lot of ANSI C macros are defined in this file. We can configure MiniGUI by enabling or disabling these macros. Generally, we can modify this file in order to configure MiniGUI. You must recompile MiniGUI if this file is modified. After that you should install the header files and the libraries on your system. If your applications are static linking to MiniGUI, you should rebuild your applications, too. Please note that you should placed the **`mgconfig.h`** in a MiniGUI header file directory which your compiler can find it and overwrite the old one.

In general, the contents of **mgconfig.h** as the follows:

```
...

/* Define if compile for VxWorks operating system */
#define __VXWORKS__ 1

/* Define if include IAL engine for ABSSIG */
/* #undef _ABSSIG_IAL */

/* Define if include IAL engine for Arca Tpanel Ads7846 */
/* #undef _ADS7846_IAL */

/* Define if include IAL engine for ADS */
/* #undef _ADS_IAL */

/* Define if include advanced 2D graphics APIs */
#define _ADV_2DAPI 1

/* Define if support Arabic charset */
/* #undef _ARABIC_SUPPORT */

/* Define if include IAL engine for ARM3000 */
/* #undef _ARM3000_IAL */

/* Define if include the automatic IAL engine */
/* #undef _AUTO_IAL */

/* Define if support BIG5 charset */
#define _BIG5_SUPPORT 1

/* Define if include clipboard support */
#define _CLIPBOARD_SUPPORT 1

...
```

Above produces is a piece of **mgconfig.h**. Macro __VXWORKS__ is defined in this file and this macro will open the VxWorks support code in the MiniGUI source code. Macro _CLIPBOARD_SUPPORT is defined in this file, too. It will open the clipboard support code. Macro _AUTO_IAL is not defined in this file and MiniGUI will not support for Auto input engine.

The attention, in **mgconfig.h** also contains other some macro definitions, for instance MiniGUI version number and so on. Please maintain these macro definitions to be invariable; do not have voluntarily to revise these macro definitions.

The handwork revises **mgconfig.h** the procedure extremely tediously, moreover is easy to make a mistake. If you use the GNU development environment, then may use the configure script to configure MiniGUI. The following section introduces how to use the configure script automatically to produce the **mgconfig.h** file in the GNU development environment.

2.1.1 Configuration in GNU Development Environment by Configure Script

It’s known that we can conveniently maintain the program package using makefile. Through makefile, we may compile, clean or install the function library, executable file and header files in the software package, etc. Although it is possible to organize a big project with makefile, it is not an easy job to create such a makefile manually. When we need to maintain a large-scale source code directory tree, the makefile maintenance work can greatly increase. Therefore, the Free Software Foundation's GNU project has developed the Autoconf/Automake tool for many software projects, which is based on the C language. Using this tool, we may automatically produce the makefile, and can check the system configuration information, which helps enhancement application software probability.

MiniGUI (MiniGUI library and sample programs package) is through the GNU Automake/Autoconf script organization. Therefore, if you use the GNU compatible development environment, for instance the Linux platform or Cygwin environment in Windows platform and so on, you may use MiniGUI’s Automake/Autoconf configuration script to configure MiniGUI. Uses MiniGUI’s Automake/Autoconf configuration script, certainly does not need to install Automake/Autoconf tool itself, but you just run the configure script in the MiniGUI source code package then to complete the configuration. If you run the configure script, it can produce not only makefile, but also **`mgconfig.h`** file base on each of option in the configure script. Afterwards, we just need run make and make install commands to compile MiniGUI, and then MiniGUI library and header files will be installed to the directory, which you assigned.

[NOTE] The MiniGUI configure script only can be used in the GNU compatible development environment. The GNU compatible development environment usually has: the Linux system, the cygwin environment running on Windows and so on, It may apply to MiniGUI-VAR product version like Linux, uClinux, eCos.

There are lot of options in the MiniGUI configure script, and each configuration option corresponds a certain macro in **`mgconfig.h`**. If you enable an option when run configure, then the correspondence macro will be defined; otherwise can’t define this macro. Run the following command.

```
user$ ./configure --help
```

You can obtain the whole options detailed list. For instance, supposing you use Debian Linux 3.1 as your development environment, the command runs in the MiniGUI source code directory and the running result as follows (this command output may have differently on other Linux release version):

```
`configure' configures this package to adapt to many kinds of systems.

Usage: ./configure [OPTION]... [VAR=VALUE]...

To assign environment variables (e.g., CC, CFLAGS...), specify them as
VAR=VALUE.  See below for descriptions of some of the useful variables.

Defaults for the options are specified in brackets.

Configuration:
  -h, --help              display this help and exit
      --help=short        display options specific to this package
      --help=recursive    display the short help of all the included packages
  -V, --version           display version information and exit
  -q, --quiet, --silent   do not print `checking...' messages
      --cache-file=FILE   cache test results in FILE [disabled]
  -C, --config-cache      alias for `--cache-file=config.cache'
  -n, --no-create         do not create output files
      --srcdir=DIR        find the sources in DIR [configure dir or `..']

Installation directories:
  --prefix=PREFIX         install architecture-independent files in PREFIX
               [/usr/local]
  --exec-prefix=EPREFIX   install architecture-dependent files in EPREFIX
               [PREFIX]

By default, `make install' will install all the files in
`/usr/local/bin', `/usr/local/lib' etc.  You can specify
an installation prefix other than `/usr/local' using `--prefix',
for instance `--prefix=$HOME'.

For better control, use the options below.

Fine tuning of the installation directories:
  --bindir=DIR           user executables [EPREFIX/bin]
  --sbindir=DIR          system admin executables [EPREFIX/sbin]
  --libexecdir=DIR       program executables [EPREFIX/libexec]
  --datadir=DIR          read-only architecture-independent data [PREFIX/share]
  --sysconfdir=DIR       read-only single-machine data [PREFIX/etc]
  --sharedstatedir=DIR   modifiable architecture-independent data [PREFIX/com]
  --localstatedir=DIR    modifiable single-machine data [PREFIX/var]
  --libdir=DIR           object code libraries [EPREFIX/lib]
  --includedir=DIR       C header files [PREFIX/include]
  --oldincludedir=DIR    C header files for non-gcc [/usr/include]
  --infodir=DIR          info documentation [PREFIX/info]
  --mandir=DIR           man documentation [PREFIX/man]

Program names:
  --program-prefix=PREFIX            prepend PREFIX to installed program names
  --program-suffix=SUFFIX            append SUFFIX to installed program names
  --program-transform-name=PROGRAM   run sed PROGRAM on installed program names

System types:
  --build=BUILD     configure for building on BUILD [guessed]
  --host=HOST       cross-compile to build programs to run on HOST [BUILD]
  --target=TARGET   configure for building compilers for TARGET [HOST]

Optional Features:
  --disable-FEATURE       do not include FEATURE (same as --enable-FEATURE=no)
  --enable-FEATURE[=ARG]  include FEATURE [ARG=yes]
  --enable-shared=PKGS  build shared libraries default=yes
  --enable-static=PKGS  build static libraries default=yes
  --enable-fast-install=PKGS  optimize for fast installation default=yes
  --disable-libtool-lock  avoid locking (might break parallel builds)
  --disable-dependency-tracking Speeds up one-time builds
  --enable-dependency-tracking  Do not reject slow dependency extractors
  --enable-ownmalloc       use own implementation of malloc functions <default=no>
  --enable-ownstdio        use own implementation of stdio functions <default=no>
  --enable-ownpthread      use own implementation of pthread functions <default=no>
  --enable-procs           build MiniGUI-Processes version <default=no>
  --enable-standalone      build MiniGUI-Standalone version <default=no>
  --enable-incoreres       use incore resource instead file IO to initialize MiniGUI <default=no
>
  --enable-miniguientry    use minigui_entry function in MiniGUI <default=no>
  --enable-fixedmath       include fixed math routines <default=yes>
  --enable-debug           build with debugging messages <default=no>
  --enable-tracemsg        trace messages of MiniGUI <default=no>
  --enable-msgstr          include symbol name of message <default=no>
  --enable-micemoveable    user can move window by using mouse <default=yes>
  --enable-dblclk          mouse button can do double click <default=yes>
  --enable-cursor          include cursor support <default=yes>
  --enable-clipboard       include clipboard support <default=yes>
  --enable-coortrans_cw    support clockwise rotation of screen in the native FB GAL engine <def
ault=no>
  --enable-coortrans_ccw   support counterclockwise rotation of screen in the native FB GAL engi
ne <default=no>
  --enable-ep7211ial       build the IAL engine for EP7211-based board <default=no>
  --enable-adsial          build the IAL engine for ADS Graphics Client board <default=no>
  --enable-ipaqial         build the IAL engine for iPAQ H3600 <default=no>
  --enable-l7200ial        build the IAL engine for L7200 <default=no>
  --enable-arm3000ial      build the IAL engine for ARM3000 <default=no>
  --enable-dm270ial        build the IAL engine for DM270 <default=no>
  --enable-evmv10ial       build the IAL engine for xScale EVMV 1.0 <default=no>
  --enable-fxrm9200ial     build the IAL engine for FXRM9200 <default=no>
  --enable-abssigial       build the IAL engine for ABSSIG <default=no>
  --enable-embest2410ial   build the IAL engine for EMBEST ARM2410 <default=no>
  --enable-embest44b0ial   build the IAL engine for EMBEST ARM44B0 <default=no>
  --enable-fft7202ial      build the IAL engine for FFT7202 <default=no>
  --enable-palm2ial        build the IAL engine for OKWAP Palm2 <default=no>
  --enable-mpc823ial       build the IAL engine for mpc823 <default=no>
  --enable-ucb1x00ial      build the IAL engine for ucb1x00 <default=no>
  --enable-px255bial       build the IAL engine for px255b <default=no>
  --enable-vr4181ial       build the IAL engine for NEC VR4181 debug board <default=no>
  --enable-helioial        build the IAL engine for Helio Touch Panel <default=no>
  --enable-tfstbial        build the IAL engine for Tongfang STB <default=no>
  --enable-t800ial         build the IAL engine for MT T800 <default=no>
  --enable-mc68x328ial     build the IAL engine for uClinux touch screen palm/mc68ez328 <default
=no>
  --enable-smdk2410ial     build the IAL engine for SMDK2410 touch screen <default=no>
  --enable-hh2410r3ial     build the IAL engine for HHARM2410R3 touch screen <default=no>
  --enable-c33l05ial       build the IAL engine for EPSONC33L05 touch screen <default=no>
  --enable-hh2440ial       build the IAL engine for HHARM2440 touch screen <default=no>
  --enable-skyeyeep7312ial build the IAL engine for touch screen of SkyEye EP7312 simulation <de
fault=no>
  --enable-dmgstbial       build the IAL engine for Intel DMG STB remote controller <default=no>
  --enable-fipial          build the IAL engine for EM85xx Front Panel and Remote Controller <de
fault=no>
  --enable-svpxxial        build the IAL engine for WinBond SVPXX <Default=no>
  --enable-ads7846ial      build the IAL engine for Arca Tpanel Ads7846 <Default=no>
  --enable-dummyial        build the Dummy IAL engine <default=yes>
  --enable-em8620ial       build the IAL engine for EM8620 <default=no>
  --enable-em86ial         build the Em86 IAL engine <default=no>
  --enable-em85ial         build the Em85 IAL engine <default=no>
  --enable-autoial         build the Automatic IAL engine <default=no>
  --enable-randomial       build the Random IAL engine <default=no>
  --enable-hhkbdirial      build the IAL engine for HHCF5249 PS/2 keyboard and IrDA <default=no>
  --enable-commial         build the COMM IAL engine <default=no>
  --enable-qvfbial         build the QVFB IAL engine <default=yes>
  --enable-wvfbial         build the WVFB IAL engine <default=no>
  --enable-utpmcial        build the IAL engine for UTPMC <default=no>
  --enable-nativeial       build the native (Linux console) IAL engine <default=yes>
  --enable-dfbial          build the IAL engine for DirectFB <default=no>
  --enable-hi3510ial         build the HI3510 IAL engine <default=no>
  --enable-nativeps2       build the native engine subdriver for PS2 mouse <default=yes>
  --enable-nativeimps2     build the native engine subdriver for IntelligentMouse (IMPS/2) mouse
<default=yes>
  --enable-nativems        build the native engine subdriver for old MS serial mouse <default=ye
s>
  --enable-nativems3       build the native engine subdriver for MS3 mouse <default=yes>
  --enable-nativegpm       build the native engine subdriver for GPM daemon <default=yes>
  --enable-textmode        Linux system have console (text mode) on FrameBuffer <default=yes>
  --enable-rbfsupport      include raw bitmap font support <default=yes>
  --enable-rbf16           include incore RBF font of ISO8859-1 8x16 fixed font (incore resource
) <default=no>
  --enable-rbf24           include incore RBF font of ISO8859-1 12x24 fixed font (incore resourc
e) <default=no>
  --enable-rbfgb12         include incore RBF font of GB2312 12x12 fixed/song font (incore resou
rce) <default=yes>
  --enable-rbfgb16         include incore RBF font of GB2312 16x16 fixed/song font (incore resou
rce) <default=no>
  --enable-rbfgb24         include incore RBF font of GB2312 24x24 fixed/song font (incore resou
rce) <default=no>
  --enable-rbfgb16hei      include incore RBF font of GB2312 16x16 fixed/hei font (incore resour
ce) <default=no>
  --enable-rbfbig12        include incore RBF font of BIG5 12x12 fixed/ming font (incore resourc
e) <default=no>
  --enable-rbfbig16        include incore RBF font of BIG5 16x16 fixed/ming font (incore resourc
e) <default=no>
  --enable-rbfbig24        include incore RBF font of BIG5 24x24 fixed/ming font (incore resourc
e) <default=no>
  --enable-rbfkj12        include incore RBF font of SHIFT-JIS Kanji fonts 12 (incore resource)
<default=no>
  --enable-rbfkj14        include incore RBF font of SHIFT-JIS Kanji fonts 14 (incore resource)
<default=no>
  --enable-rbfkj16        include incore RBF font of SHIFT-JIS Kanji fonts 16 (incore resource)
<default=no>
  --enable-vbfsupport      include var bitmap font support <default=yes>
  --enable-fontsserif      include incore font sansserif <default=yes>
  --enable-fontcourier     include incore font courier <default=yes>
  --enable-fontsymbol      include incore font symbol <default=no>
  --enable-fontvgas        include incore font VGAs <default=no>
  --enable-fonthelv        include incore font Helvetica <default=no>
  --enable-qpfsupport      build support for Qt Prerendered Font (QPF) <default=yes>
  --enable-ft2support      build support for FreeType2 library <default=no>
  --enable-ttfsupport      build support for TrueType font <default=no>
  --enable-ttfcache        include ttf cache support <default=no>
  --enable-type1support    build support for Adobe Type1 font <default=yes>
  --enable-latin2support   include East European (Latin 2, ISO-8859-2) charset support <default=
no>
  --enable-latin3support   include South European (Latin 3, ISO-8859-3) charset support <default
=no>
  --enable-latin4support   include North European (Latin 4, ISO-8859-4) charset support <default
=no>
  --enable-cyrillicsupport include Cyrillic (ISO-8859-5) charset support <default=no>
  --enable-arabicsupport   include Arabic (ISO-8859-6) charset support <default=no>
  --enable-greeksupport    include Greek (ISO-8859-7) charset support <default=no>
  --enable-hebrewsupport   include Hebrew (ISO-8859-8) charset support <default=no>
  --enable-latin5support   include Turkish (Latin 5, ISO-8859-9) charset support <default=no>
  --enable-latin6support   include Nordic, Latin 6, ISO-8859-10) charset support <default=no>
  --enable-thaisupport     include Thai (ISO-8859-11) charset support <default=no>
  --enable-latin7support   include Latin 7 (ISO-8859-13) charset support <default=no>
  --enable-latin8support   include Latin 8 (ISO-8859-14) charset support <default=no>
  --enable-latin9support   include Latin 9 (ISO-8859-15, West Extended) charset support <default
=yes>
  --enable-latin10support  include Latin 10 (ISO-8859-16, Romanian) charset support <default=no>
  --enable-gbsupport       include EUC encoding of GB2312 charset support <default=yes>
  --enable-gbksupport      include GBK charset support <default=yes>
  --enable-gb18030support  include GB18030-0 charset support <default=no>
  --enable-big5support     include BIG5 charset support <default=yes>
  --enable-euckrsupport    include support for EUC encoding of KSC5636 and KSC5601 charsets <def
ault=no>
  --enable-eucjpsupport    include support for EUC encoding of JISX0201 and JISX0208 charsets <d
efault=no>
  --enable-shiftjissupport include support for Shift-JIS encoding of JISX0201 and JISX0208 chars
ets <default=no>
  --enable-unicodesupport  include UNICODE (ISO-10646-1 and UTF-8 encoding) support <default=yes
>
  --enable-kbdfrpc         include keyboard layout for French PC keyboard (non-US 102 keys) <def
ault=no>
  --enable-kbdfr           include keyboard layout for French <default=no>
  --enable-kbdde           include keyboard layout for German <default=no>
  --enable-kbddelatin1     include keyboard layout for German Latin1 <default=no>
  --enable-kbdit           include keyboard layout for Italian <default=no>
  --enable-kbdes           include keyboard layout for Spanish <default=no>
  --enable-kbdescp850      include keyboard layout for Spanish CP850 <default=no>
  --enable-savebitmap      include SaveBitmap-related functions <default=yes>
  --enable-pcxsupport      include PCX file support <default=no>
  --enable-lbmsupport      include LBM/PBM file support <default=no>
  --enable-tgasupport      include TGA file support <default=no>
  --enable-gifsupport      include GIF file support <default=yes>
  --enable-jpgsupport      include JPG file support <default=yes>
  --enable-pngsupport      include PNG file support <default=yes>
  --enable-imegb2312       include IME (GB2312) support <default=yes>
  --enable-imegb2312py     include IME (GB2312) Intelligent Pinyin module <default=yes>
  --enable-mousecalibrate  include code doing mouse calibration <default=yes>
  --enable-aboutdlg        include About Dialog Box <default=yes>
  --enable-savescreen      include code for screenshots <default=yes>
  --enable-grayscreen      target is a gray screen <default=no>
  --enable-tinyscreen      target is a tiny-size screen <default=no>
  --enable-fixedbarlen     scrollbar has a fixed bar length <default=no>
  --enable-ctrlstatic      include STATIC control <default=yes>
  --enable-ctrlbutton      include BUTTON control <default=yes>
  --enable-ctrlsimedit     include Simple EDIT control <default=yes>
  --enable-ctrlsledit      include Single-Line EDIT control <default=yes>
  --enable-ctrllistbox     include LISTBOX control <default=yes>
  --enable-ctrlpgbar       include PROGRESSBAR control <default=yes>
  --enable-ctrltoolbar     include TOOLBAR control <default=yes>
  --enable-ctrlnewtoolbar  include NEWTOOLBAR control <default=yes>
  --enable-ctrlmenubtn     include MENUBUTTON control <default=yes>
  --enable-ctrltrackbar    include TRACKBAR control <default=yes>
  --enable-ctrlcombobox    include COMBOBOX control <default=yes>
  --enable-ctrlpropsheet   include PROPSHEET control <default=yes>
  --enable-ctrlscrollview  include SCROLLVIEW and SCROLLWINDOW controls <default=yes>
  --enable-ctrltextedit    include TEXTEDIT control which is based-on SCROLLVIEW control <defaul
t=yes>
  --enable-extctrlmonthcal     include MONTHCALENDAR control in MiniGUIExt library <default=yes>
  --enable-extctrltreeview     include TREEVIEW control in MiniGUIExt library <default=yes>
  --enable-extctrlspinbox      include SPINBOX control in MiniGUIExt library <default=yes>
  --enable-extctrlcoolbar      include COOLBAR control in MiniGUIExt library <default=yes>
  --enable-extctrllistview     include LISTVIEW control in MiniGUIExt library <default=yes>
  --enable-extctrliconview     include ICONVIEW control in MiniGUIExt library <default=yes>
  --enable-extctrlgridview    include GRIDVIEW control in MiniGUIExt library (test) <default=no>
  --enable-extctrlanimation    include ANIMATION control and GIF87a/GIF89a support in MiniGUIExt
library <default=yes>
  --enable-extskin             include skin support in MiniGUIExt library <default=yes>
  --enable-libvcongui          include support for Virtual Console on Linux <default=yes>
  --enable-videodummy          include dummy NEWGAL engine <default=yes>
  --enable-videofbcon          include FrameBuffer console NEWGAL engine <default=yes>
  --enable-videoqvfb           include Qt Virtual FrameBuffer NEWGAL engine <default=yes>
  --enable-videowvfb           include windows Virtual FrameBuffer NEWGAL engine <default=no>
  --enable-videocommlcd        include NEWGAL engine for Common LCD <default=no>
  --enable-videoshadow         include Shadow NEWGAL engine <default=no>
  --enable-videoem86gfx        include NEWGAL engine for EM86xx GFX <default=no>
  --enable-videoem85xxyuv      include NEWGAL engine for EM85xx YUV <default=no>
  --enable-videoem85xxosd      include NEWGAL engine for EM85xx OSD <default=no>
  --enable-videosvpxxosd       include SVPXXOSD NEWGAL engine <default=no>
  --enable-videobf533          include NEWGAL engine for BF533 OSD via SPI <default=no>
  --enable-videomb93493    include NEWGAL engine for mb93493 YUV FrameBuffer driver <default=no>
  --enable-videoutpmc          include NEWGAL engine for UTPMC <default=no>
  --enable-videodfb            include NEWGAL engine for DirectFB <default=no>
  --enable-videohi35xx           include Hi35xx Video  NEWGAL engine <default=no>
  --enable-adv2dapi            include advanced 2D graphics APIs <default=yes>

Optional Packages:
  --with-PACKAGE[=ARG]    use PACKAGE [ARG=yes]

  --without-PACKAGE       do not use PACKAGE (same as --with-PACKAGE=no)
  --with-gnu-ld           assume the C compiler uses GNU ld default=no
  --with-pic              try to use only PIC/non-PIC objects default=use both
--with-osname=linux/uclinux/ecos/ucos2/swlinux/vxworks/win32/darwin/threadx/cygwin/nucleus/ose/p
sos
  --with-targetname=unknown/stb810/vfanvil/vxi386/qvfb/fbcon/mx21/monaco/c33l05/bfin/vxppc
  --with-style=classic/flat/fashion
--with-ttfcachesize=64/128/256/512/1024
--with-mttfcachenum=10/20/40

Some influential environment variables:
  CC          C compiler command
  CFLAGS      C compiler flags
  LDFLAGS     linker flags, e.g. -L<lib dir> if you have libraries in a
              nonstandard directory <lib dir>
  CPPFLAGS    C/C++ preprocessor flags, e.g. -I<include dir> if you have
              headers in a nonstandard directory <include dir>
  CPP         C preprocessor

Use these variables to override the choices made by `configure' or to help
it to find libraries and programs with nonstandard names/locations.
```

Above these parameters were already configured parameter which established in the configure script, and these parameters are allowed to control which function codes were supported when compile MiniGUI. For example, run:

```
user$ ./configure --with-style=fashion --enable-procs --enable-standalone
```

You may configure MiniGUI that is the Fashion style and the MiniGUI-Standalone runtime mode. If you run:

```
user$ ./configure --with-style=flat  --enable-procs
```

Then configure MiniGUI that is the MiniGUI-Processes runtime mode and the Flat style. Without any parameter execution **`./configure`** command will produce a Makefile with default configuration options. Each compiling configuration option has provided a default setting in its explanation: <**`default=yes`**> (Expressed this compiling configuration option is enabled default) or <**`default=no`**> (Expressed this compiling configuration option is disabled default).

Besides the MiniGUI has defined configuration options, the configure script also has some important general compiling configuration options.

1) Prefix Option

This compiling configuration option assigns the MiniGUI library where to install. The default installation path is **`/usr/local`**. If you run:

```
user$ ./configure --prefix=/home/test
```

After executing 'make install' command, the function library, header files and reference document are installed in **`/home/test/lib`**, **`/home/test/include`** and **`/home/test/man`** directory.

2) Cross Compiling Option

The compiling configuration options **`--build`**, **`--host`** and **`--target`** are very important to cross compile applications. For example, if you use the arm-linux cross compiling toolchain, then you may assign option like **--build**, thus let the configure script produces the makefile file used to complete the arm-linux's cross compiling:

```
user$ CC=arm-linux-gcc ./configure --prefix=/usr/local/arm/2.95.3/arm-linux/ \
    --build=i386-linux \
    --host=arm-unknown-linux \
    --target=arm-unknown-linux
```

In above command, the **--prefix** option is used to set the installing MiniGUI configuration file, the function library and the header files directory's prefix, when you executed **make install** command, MiniGUI configuration file, the library file and header files will be installed in the following position:

- **/usr/local/arm/2.95/arm-linux/etc/**
- **/usr/local/arm/2.95.3/arm-linux/lib/**
- **/usr/local/arm/2.95.3/arm-linux/include/**

3) --enable-static and --enable-shared

The two configuration options assign whether generating static function library or dynamic function library. If you do not need to produce the static library, then you may use the **--disable-static** configuration option, it will take less time to compile the library than default.

There are several predefined targets in the makefile, which produced by the configure script supply for user, here only everal summaries as follow:

The function storehouse, a document and so on are installed in the way, which assigns

- make all: Produce the target set. Only run make command also to be allowed, this time can start to compile the source code, then link it and produces the executable file or function library.
- make clean: Clean the previous object files(*.o).
- make install: Install the function library, header files and so on to the directory which you assigned.

2.1.2 Configuration under Non-GNU environment

A majority of traditional embedded operating system supported by MiniGUI, user usually can use the integrated development environment running on Windows platform, such as Tornado, ADS, etc. Because these environment provide the development tools chain that is not GNU compatible, therefore, we are unable to use the configure script that is described in section 2.1.1 to produce makefile and the `mgconfig.h` file automatically. In this kind of situation, we need voluntarily to revise the `mgconfig.h` file to complete the MiniGUI compiling configuration. Fortunately, Feynman Software already prepares the `mgconfig.h` file for the majority operating system, which can directly be used (store in MiniGUI source code `build/` directory); moreover Feynman Software also prepared the corresponding development environment project file. You may directly manually revise the `mgconfig.h` file based on these project environments, and compile the MiniGUI library. For more detail information, please refer to the section 2.4.2.

2.2 Detailed Description on Compiling, Configuration Options of MiniGUI

In this chapter, we will give detailed description on all compiling, configuration options of MiniGUI. MiniGUI has many compiling, configuration options, for your actual demand; you can combine these options to generate MiniGUI function library.

In GNU development environment, we implement the most of configuration options of MiniGUI that based on `--disable-FEATURE` and `--enable-FEATURE`, while MiniGUI configuration script also provides `--with-configuration` option, you can use this configuration option to choose one option from multiple specified configuration. For example, you can use `--with-style` configuration option to specify the style of window and control of MiniGUI. Finally, these configuration options were defined macros, whatever use `--disable-FEATURE` or `--enable-FEATURE` or `--with-configuration` option to specify configuration option.

In the next chapter, we will give configuration option of MiniGUI by classify. We will description on configuration names of configure script and macro names in the `mgconfig.h` file.

2.2.1 Operating System Options and Macros

MiniGUI provides support for multiple operating systems, you can specify operating system when execute configure script, default operating system is Linux. If you want to run MiniGUI on uClinux, you can execute command as the follow:

```
user$ ./configure  --with-osname=uclinux
```

If you specify an operating system, the corresponding macro was defined in **`mgconfig.h`**. For some operating systems, we will open other some macros. Table 2.1 lists relevant options and macros of operating systems.

Table 2.1 operating systems relevant options and macros

Configuration options	Macro	Other relevant macro	Memo
--with-osname=linux	__LINUX__		Default value, for Linux operating system
--with-osname=uclinux	__uClinux__		For uClinux operating system
--with-osname=ecos	__ECOS__	__NOUNIX__	For eCos operating system
--with-osname=ucos2	__UCOSII__	__NOUNIX__ _INCORE_RES _USE_OWN_MALLOC _USE_OWN_STDIO _USE_OWN_PTHREAD	For uC/OS-II operating system
--with-osname=swlinux	__WINBOND_SWLINUX__		For SWLinux operating system, mutation of uClinux operating system
--with-osname=vxworks	__VXWORKS__	__NOUNIX__ _USE_OWN_STDIO _USE_OWN_PTHREAD	For VxWorks operating system
--with-osname=cygwin	__CYGWIN__	__NOUNIX__	For cygwin environment
--with-osname=win32	WIN32	__NOUNIX__	For Win32 platform
--with-osname=darwin	__DARWIN__	__NOUNIX__	For MacOS X operating system
--with-osname=threadx	__THREADX__	__NOUNIX__ _INCORE_RES _USE_OWN_MALLOC _USE_OWN_STDIO _USE_OWN_PTHREAD	For ThreadX operating system
--with-osname=nucleus	__NUCLEUS__	__NOUNIX__ _INCORE_RES _USE_OWN_MALLOC _USE_OWN_STDIO _USE_OWN_PTHREAD	For Nucleus operating system
--with-osname=ose	__OSE__	__NOUNIX__ _INCORE_RES _USE_OWN_PTHREAD	For OSE operating system
--with-osname=psos	__PSOS__	__NOUNIX__ _INCORE_RES _USE_OWN_PTHREAD	For pSOS operating system

According to operating system, we divide MiniGUI value-added release, so the MiniGUI value-added release product for certain operating system cannot run on anther operating system. In order to run MiniGUI value-added release product on corresponding operating system, you make sure that the above macros were defined when you modify configuration.

2.2.2 Target Board Related Options and Macros

In MiniGUI certain codes are related with a special target board; if you want run MiniGUI must on these target boards correctly, you need to assign the name of these development boards. When you run configure script, through the `--with-targetname` option, may assign the special target board name and the default name is unknown. The target board related options usually use for assign the sub-driver of graphics engine when MiniGUI uses the Shadow graphics engine or the CommLCD graphics engine, in other words, when uses these two engines, through the target board name you can determine which sub-driver contains. The table 2.2 lists the target board related options and macros.

Table 2.2 target board related options and macros

Configuration options	Macro	Memo
--with-targetname=stb810	__TARGET_STB810__	Philips STB810 development board base on Linux
--with-targetname=vfanvil	__TARGET_VFANVIL__	VisualFone development board base on ThreadX
--with-targetname=vxi386	__TARGET_VXI386__	i386 target base on VxWorks
--with-targetname=qvfb	__TARGET_QVFB__	Include qvfb sub-driver of Shadow engine base on Linux
--with-targetname=wvfb	__TARGET_WVFB__	Include wvfb sub-driver of Shadow engine base on Windows
--with-targetname=fbcon	__TARGET_FBCON__	Include fbcon sub-driver of Shadow engine base on Linux
--with-targetname=mx21	__TARGET_MX21__	MX21 development board base on OSE
--with-targetname=c33l05	__TARGET_C33L05__	Epson C33L05 development board base on axLinux
--with-targetname=bfin	__TARGET_BLACKFIN__	BlackFin537 development board base on uClinux
--with-targetname=vxppc	__TARGET_PPC__	PowerPC target base on VxWorks
--with-targetname=monaco	__TARGET_MONACO__	monaco development board base on Nucleus
--with-targetname=unkown	__TARGET_UNKNOWN__	Unknown development board: default value

2.2.3 Runtime Mode Related Options and Macros

We can configure MiniGUI as one of three kind of runtime mode: MiniGUI-Processes runtime mode base on multi-processes, MiniGUI-Threads runtime mode base on multi-thread, as well as MiniGUI-Standalone runtime mode base on non-multi-processes also non-multi-thread. MiniGUI-Threads runtime mode is the default mode when MiniGUI use the default configuration option. The table 2.3 lists runtime mode related options and macros.

Table 2.3 runtime mode related options and macros

Configuration options	Macro	Memo	Default
not assigned	_MGRM_THREADS	MiniGUI-Threads runtime mode	
procs	_MGRM_PROCESSES _LITE_VERSION	MiniGUI-Processes runtime mode, support Linux/uClinux operating system only	Disabled
standalone	_MGRM_STANDALONE _LITE_VERSION _STAND_ALONE	MiniGUI-Standalone runtime mode, support Linux/uClinux operating system only	Disabled

MiniGUI-VAR for uClinux V1.6.10 product is not support MiniGUI-Processes runtime mode. The table 2.4 lists runtime mode related options and macros of MiniGUI-VAR for uClinux V1.6.10 product.

Table 2.4 runtime mode related options and macros of MiniGUI-VAR for Linux/uClinux V1.6.10 product

Configuration options	Macro	Memo	Default
not assigned		MiniGUI-Threads runtime mode	
standalone	_LITE_VERSION _STAND_ALONE	MiniGUI-Standalone runtime mode, support Linux/uClinux operating system only, needs enable lite option	Disabled

2.2.4 Graphics Engine Related Options and Macros

MiniGUI supports many kinds of graphics engine. The commonly used graphics engine mainly includes the Dummy graphics engine, Qt Virtual FrameBuffer engine, Linux FrameBuffer console graphics engine, the COMMLCD graphics engine, the Shadow graphics engine, Windows Virtual FrameBuffer graphics engine and so on. Through the configuration option or macro, we may contain a certain graphics engine to MiniGUI. But if you assign MiniGUI to use a certain graphics engine, then you need to assign a special runtime configuration option. For instance, if you assign MiniGUI to use the dummy graphics engine, you may assign the runtime configuration option `gal_engine=dummy` in [`system`] section, the graphics engine name is on the right of the equal sign. The attention, the engine name is case sensitivity. About how to revises the runtime configuration option, please refer the 3rd chapter of *MiniGUI Runtime Configuration Options* this handbook. The table 2.5 lists the graphics engine related options, macros and the name.

Table 2.5 graphics engine related options and macros

Configuration options	Macro	Engine name	Memo	Default
videodummy	_NEWGAL_ENGINE_DUMMY	dummy	All operating system	Enabled
videofbcon	_NEWGAL_ENGINE_FBCON	fbcon	Linux/uClinux	Enabled
videoqvfb	_NEWGAL_ENGINE_QVFB	qvfb	Linux	Enabled
videowvfb	_NEWGAL_ENGINE_WVFB	wvfb	Win32	Disabled
videocommlcd	_NEWGAL_ENGINE_COMMLCD	commlcd	All operating system	Disabled
videoshadow	_NEWGAL_ENGINE_SHADOW	shadow	All operating system, MiniGUI-Threads , MiniGUI-Standalone runtime mode	Disabled
videoem86gfx	_NEWGAL_ENGINE_EM86GFX	em86gfx	GFX graphics engine of EM86xx development board	Disabled
videoem85xxosd	_NEWGAL_ENGINE_EM85XXOSD	em85xxosd	OSD graphics engine of EM85xx development board, uClinux	Disabled
videoem85xxyuv	_NEWGAL_ENGINE_EM85XXYUV	em85xxyuv	YUV graphics engine of EM85xx development board, uClinux	Disabled
videosvpxxosd	_NEWGAL_ENGINE_SVPXXOSD	svpxxosd	uClinux	Disabled
videobf533	_NEWGAL_ENGINE_BF533	bf533	uClinux	Disabled
videomb93493	_NEWGAL_ENGINE_MB93493	mb93493	uCinux	Disabled
videoutpmc	_NEWGAL_ENGINE_UTPMC	utpmc	uClinux	Disabled
Videohi35xx	_NEWGAL_ENGINE_HI3510 _NEWGAL_ENGINE_HI3560	hi35xx	graphics engine of hi35xx development board, Linux	Disabled
videodfb	_NEWGAL_ENGINE_DFB	dfb	Run MiniGUI on DirectFB, Linux	Disabled

The Dummy is a graphics engine ("mute" graphics engine), which it does not make any actual output. Therefore, if the graphics engine for your development board still cannot work, you can run MiniGUI using this graphics engine.

The Qvfb graphics engine uses in the Linux operating system. Using qvfb, we can run the MiniGUI program in X Window; it may greatly facilitate the application debugging. Similar with the qvfb graphics engine, when uses MiniGUI SDK for Win32 run MiniGUI program on Win32 platform, it run on Windows Virtual in the FrameBuffer actually, and use the wvfb graphics engine.

In MiniGUI also has a special Shadow graphics engine, uses the Shadow graphics engine, MiniGUI may support the graphic display devices which it is lower than 8 bit colors, also support the screen rotation. The Shadow graphics engine has used the sub-driver concept; it determined which sub-driver contains through the target board name. Only one sub-driver can be contained at one time, it determined by the target board configuration option (sees section 2.2.2). The attention, the Shadow graphics engine is disabled as the default; moreover it is only suitable for the MiniGUI-Threads and MiniGUI-Standalone runtime mode at present.

The sub-drivers of the Shadow graphics in MiniGUI are (in MiniGUI source code directory **`src/newgal/shadow`**):

- **`unknown`**: the default sub-driver, similar with the dummy graphics engine, user may modify this sub-driver in order to operate and visit the low graphics devices.
- **`vfanvil`**: sub-driver for VisualFone Anvil development board. For ThreadX operating system.
- **`qvfb`**: sub-driver for Linux QVFB all display mode, support low than 8-bit color display mode and screen rotation.
- **`fbcon`**: sub-driver for Linux console FrameBuffer, support low than 8-bit color display mode and screen rotation.
- **`wvfb`**: sub-driver for Windows Virtual FrameBuffer(wvfb), support low than 8-bit color display mode and screen rotation.

We can rotate the screen by Shadow engine. Table 2.6 lists the screen rotation related options and macros.

Table 2.6 screen rotation related options and macros

Configuration options	Macro		Macro value	Comment	Default
coortrans_cw	_COOR_TRANS	_ROT_DIR_CW	1	Rotate screen clockwise	Disabled
coortrans_ccw		_ROT_DIR_CW	0	Rotate screen anticlockwise	Disabled

The CommLCD graphics engine is the most used graphics engine when MiniGUI run on the tradition embedded operating system. CommLCD also uses the sub-driver structure like Shadow graphics engine. At present, sub-drivers for CommLCD graphics engine are:

- **`vxi386`**: Sub-driver for VxWorks i386 target board.
- **`unknown`**: Sub-driver implemented by the eCos interface standard if use eCos operating system. Otherwise, user should define the sub-driver. There is a implementation for uC/OS-II operating system in MiniGUI source code directory

include/mgdrv-ucosii.c. You can revise this file in order to support your own LCD controller.

2.2.5 Input Engine Related Options and Macros

MiniGUI provides some input engine, which can be used directly for many kinds of development board. Generally the input engines include the Dummy input engine, Qt Virtual FrameBuffer engine, Linux FrameBuffer console input engine, the COMM input engine, the Random input engine, Windows Virtual FrameBuffer input engine and so on. Through the configuration options or macros, we can contain an input engine to MiniGUI. But if assign MiniGUI to use a certain input engine, then you need to assign a special runtime configuration option. For instance, If you assign MiniGUI to use the dummy input engine, you may assign the runtime configuration option **ial_engine=dummy** in [**system**] section, the input engine name is on the right of the equal sign. The attention, the engine name is case sensitivity. About how to revises the runtime configuration option, please refer the 3rd chapter of *MiniGUI Runtime Configuration Options* this handbook. The table 2.7 lists the input engine related options and macros.

Table 2.7 input engines related options and macros

Configuration options	Macro	Engine name	Comment	Default
dummyial	_DUMMY_IAL	dummy	Dummy input engine, for all operating system	Enabled
autoial	_AUTO_IAL	auto	Automatic input engine, for all operating system	Disabled
nativeial	_NATIVE_IAL_ENGINE	console	Linux console input engine, Linux	Enabled
qvfbial	_QVFB_IAL	qvfb	QVFB input engine, Linux, use QVFB graphics engine	Enabled
randomial	_RANDOM_IAL	random	Random input engine, for all operating system	Disabled
wvfbial	_WVFB_IAL	wvfb	WVFB input engine, Win32, use WVFB graphics engine	Disabled
commial	_COMM_IAL	comm	COMM input engine, for all operating system	Disabled
dfbial	_DFB_IAL	dfb	Base on DirectFBinput engine, Linux, use DFB graphics engine	Disabled
ep7211ial	_EP7211_IAL	EP7211	For EP7211 board input engine	Disabled
adsial	_ADS_IAL	ADS	For ADS Graphics Client board input engine	Disabled
ipaqial	_IPAQ_IAL	ipaq	Construct input engine for iPAQ H3600	Disabled
l7200ial	_L7200_IAL	l7200	Construct input engine for L7200	Disabled

arm3000ial	_ARM3000_IAL	arm3000	Construct input engine for ARM3000	Disabled
dm270ial	_DM270_IAL	dm270	Construct input engine for DM270	Disabled
evmv10ial	_EVMV10_IAL	evmv10	xScale EVMV 1.0 input engine	Disabled
fxrm9200ial	_FXRM9200_IAL	fxrm9200	FXRM9200 input engine	Disabled
abssigial	_ABSSIG_IAL	abssig	ABSSIG input engine	Disabled
embest2410ial	_EMBEST2410_IAL	embest2410	EMBEST ARM2410 input engine	Disabled
embest44b0ial	_EMBEST44B0_IAL	embest44b0	EMBEST ARM44B0 input engine	Disabled
fft7202ial	_FFT7202_IAL	fft7200	FFT7202 input engine	Disabled
palm2ial	_PALMII_IAL	palm2	OKWAP Palm2 input engine	Disabled
mpc823ial	_MPC823_IAL	MPC823	mpc823 input engine	Disabled
ucb1x00ial	_UCB1X00_IAL	UCB1X00	ucb1x00 input engine	Disabled
px255bial	_PX255B_IAL	PX255B	px255b input engine	Disabled
vr4181ial	_VR4181_IAL	VR4181	NEC VR4181 input engine	Disabled
helioial	_HELIO_IAL	Helio	Helio Touch Panel input engine	Disabled
tfstbial	_TFSTB_IAL	TF-STB	Tongfang STB input engine	Disabled
t800ial	_T800_IAL	T800	MT T800 input engine	Disabled
mc68x328ial	_MC68X328_IAL	MC68X328	Palm/mc68ez328 uClinux touch screen input engine	Disabled
smdk2410ial	_SMDK2410_IAL	SMDK2410	SMDK2410 touch screen input engine	Disabled
hh2410r3ial	_HH2410R3_IAL	hh2410r3	HHARM2410R3 touch screen input engine	Disabled
c33l05ial	_C33L05_IAL	C33l05	EPSONC33L05 touch screen input engine	Disabled
hh2440ial	_HH2440_IAL	hh2440	HHARM2440 touch screen input engine	Disabled
skyeyeep7312ial	_SKYEYE_EP7312_IAL	SkyEyeEP7312	SkyEye EP7312 simulator touch screen input engine	Disabled
dmgstbial	_DMGSTB_IAL	dmg-stb	Intel DMG STB remote controller input engine	Disabled
fipial	_FIP_IAL	fip	EM85xx front board and remote controller input engine	Disabled
svpxxial	_SVPXX_IAL	svpxx	WinBond SVPXX input engine	Disabled
ads7846ial	_ADS7846_IAL	ads7846	Arca Tpanel Ads7846 input engine	Disabled
em8620ial	_EM8620_IAL	em8620	Em8620 development board input engine	Disabled
em86ial	_EM86_IAL	em86	Em86XX development board input engine	Disabled
em85ial	_EM85_IAL	em85	Em85XX development board input engine	Disabled
hi3510ial	_HI3510_IAL	hi3510	hi35XX development board input engine	Disabled
hhkbdirial	_HH5249KBDIR_IAL	hh5249kbdir	Input engine for HHCF5249 PS/2 keyboard and IrDA	Disabled
utpmcial	_UTPMC_IAL	utpmc	UTPMC input engine	Disabled

The Dummy input engine ("mute" input engine) is not connected to any actual input device; therefore it can’t get any input. Therefore, if the input engine for your development board still cannot to work, you can run MiniGUI using this input engine. Attention, MiniGUI use Dummy input engine when it cannot find the matched input engine in configuration options.

Like the Dummy input engine, MiniGUI provide other two input engine, which it is not associated to any device, for instance Auto input engine and Random input engine. The Auto engine may circulation produce the events automatic according the previous setting; But the Random input engine produce the random input event. These two engines may use for MiniGUI and its application software test automation.

The Console input engine aims at the PC console of Linux operating system. This input engine supports the standard PC keyboard as well as many kinds of mouse protocol. You need configure mtype and mdev field in [system] section assign the mouse protocol and the mouse device when use the console input engine.

Mouse protocol related options and macros, which console input engine supported, are listed in table 2.8. Attention, although MiniGUI support intelligence mouse, but MiniGUI does not support in the middle key and the hoop input event.

Table 2.8 Mouse protocol related options and macros

configuration options	Macro	Comment	Default
nativeps2	_PS2_SUPPORT	Support PS2 mouse protocol	Enabled
nativeimps2	_IMPS2_SUPPORT	Support intelligence mouse(IMPS/2) protocol	Enabled
nativems	_MS_SUPPORT	Support old MS serial-port mouse	Enabled
nativems3	_MS3_SUPPORT	Support MS3 mouse protocol	Enabled
nativegpm	_GPM_SUPPORT	Support GPM Daemon processes	Enabled

Except the options above, MiniGUI has also provided mouse and touch screen adjustment interfaces for applications. If you want to use this interfaces, you need to open the option about touch screen adjusts. The table 2.9 lists touch screen adjustment related options and macros.

Table 2.9 mouse and touch screen adjustment related options and macros

configuration	Macro	Comment	Default

options			
mousecalibrate	_MISC_MOUSECALIBRATE	Support touch screen adjustment	Enabled

2.2.6 Keyboard Layout Related Options and Macros

The MiniGUI keyboard layout uses for control the behavior of function TranslateMessage. Different keyboard layout will translate a same key as a different character (distinguish by the scan code). This translation process is implemented through query the scan code mapping table. At present, in MiniGUI contains the Western Europe country commonly used keyboard layout support, standard American 1.01/102 keyboard as default. If you want to use different keyboard layout in your program, you should call the function SetKeyboardLayout by the keyboard layout name. For more information, please refer *MiniGUI Programming Guide V2.0-4*. Table 2.10 listed the keyboard layout related options, macros and the name.

Table 2.10 keyboard layout related options and macros

configuration options	Macro	Keyboard layout name	Comment	Default
Kbdfrpc	_KBD_LAYOUT_FRPC	frpc	Keyboard layout for French PC keyboard (non-US 102 keys)	Disabled
Kbdfr	_KBD_LAYOUT_FR	fr	Keyboard layout for French	Disabled
Kbdde	_KBD_LAYOUT_DE	de	Keyboard layout for German	Disabled
kbddelatin1	_KBD_LAYOUT_DELATIN1	delatin1	Keyboard layout for German Latin1	Disabled
Kbdit	_KBD_LAYOUT_IT	it	Keyboard layout for Italian	Disabled
Kbdes	_KBD_LAYOUT_ES	es	Keyboard layout for Spanish	Disabled
kbdescp850	_KBD_LAYOUT_ESCP850	escp850	Keyboard layout for Spanish CP850	Disabled

2.2.7 System Global Configuration Options and Macros

The table 2.11 lists system global configuration options and macros.

Table 2.11 system global configuration options and macros

configuration options	Macro	Comment	Default
incoreres	_INCORE_RES	Use MiniGUI in-core resource	Disabled
miniguientry	_USE_MINIGUIENTRY	Use MiniGUI minigui_entry function	Disabled
debug	_DEBUG	Include debug information	Disabled
tracemsg	_TRACE_MSG	Trace MiniGUI message	Disabled

msgstr	_MSG_STRING	Include the string name of the message	Disabled
micemoveable	_MOVE_WINDOW_BY_MOUSE	Move the window by mouse	Enabled
dblclk	_DOUBLE_CLICK	Support mouse double click	Enabled
cursor	_CURSOR_SUPPORT	Support mouse cursor	Enabled
clipboard	_CLIPBOARD_SUPPORT	Support clipboard	Enabled
textmode	_HAVE_TEXT_MODE	Console base on Linux FrameBuffer	Enabled
savebitmap	_SAVE_BITMAP	Support SaveBitmap related functions	Enabled
aboutdlg	_MISC_ABOUTDLG	Include About dialog box	Enabled
savescreen	_MISC_SAVESCREEN	Support screen capture	Enabled
libvcongui	_LIB_VCONGUI	Support virtual Linux console	Enabled
fixedmath	_FIXED_MATH	Use fixed math functions	Enabled
adv2dapi	_ADV_2DAPI	Support advanced 2D graphics API	Enabled

Some important configurations are introduced as the follow:

The **incoreres** option is used to control whether MiniGUI needs fonts, bitmaps, cursors, icons and so on construct in the function library. This option is very useful for tradition embedded operating system. Because in the majority situation, the tradition embedded operating system has not file system support, supporting by the in-core resource, it was allowed to construct the above resources in the function library, and MiniGUI can run without file system. Attention in, when uses in-core resources, MiniGUI runtime configuration options can be compiled into MiniGUI library directly.

The **miniguientry** option uses for control how to implement the function MiniGUIMain. In the default situation (disabled this option), The function MiniGUIMain can be expanded to the function main, so application should not define the main function. The function MiniGUIMain can be expanded to the function **minigui_entry** when option **miniguientry** is enabled. It is easy for debug and system integration for some tradition embedded operating system.

The **fixedmath** option uses for control whether fixed math is included in MiniGUI library, such as fixcos and so on. The clipboard option uses for control whether MiniGUI is support clipboard or not; if this option is disabled, and the editor cannot support cut and copy. The **adv2api** option is control whether the MiniGUI include the advanced 2D graphics API.

The debug, **tracemsg** and **msgstr** use for MiniGUI debugging, it is not suggested user use it.

MiniGUI supports mouse cursor default. When target system has not any fix point device like mouse or touch screen, we do not need display the mouse cursor, so we can disabled the mouse cursor supporting from the configuration options.

The function Windows can be moved by mouse in MiniGUI is enabled at many embedded system, generally MiniGUI does not need cascading window user interface. Moving window is not need, too. Then **micemoveable** option can be disabled, too.

2.2.8 Input Method Related Options and Macros

There is a GB2312 input method and a GB2312 Pin-Yin input method build in MiniGUI. These two input method are enabled as the default. Table 2.12 lists the input method related options and macros.

Table 2.12 input method related options and macros

configuration options	Macro	Comment	Default
Imegb2312	_IME_GB2312	Support GB2312 input method	Enabled
Imegb2312py	_IME_GB2312_PINYIN	Support GB2312 Pin-Yin input method	Enabled

2.2.9 Character Set and Font Related Options and Macros

MiniGUI has rich support for font. It supports RBF font, VBF font (these two kinds of font are defined by MiniGUI), QPF font, TrueType font, Adobe Type1 font and so on. Because MiniGUI supports many kinds of font, so there are many flexible configuration options for font.

Like the type of font, MiniGUI provides a well support for character set. A special character set support also can be flexible configured. Table 2.13 lists character set and font related options and macros.

Table 2.13 character set and font related options and macros

configuration options	Macro	Comment	Default
latin2support	_LATIN2_SUPPORT	Include East European (Latin 2, ISO-8859-2) charset support	Disabled
latin3support	_LATIN3_SUPPORT	Include South European (Latin 3, ISO-8859-3) charset support	Disabled
latin4support	_LATIN4_SUPPORT	Include North European (Latin 4, ISO-8859-4) charset support	Disabled
cyrillicsupport	_CYRILLIC_SUPPORT	Include Cyrillic (ISO-8859-5) charset support	Disabled
arabicsupport	_ARABIC_SUPPORT	Include Arabic (ISO-8859-6)	Disabled

		charset support	
greeksupport	_GREEK_SUPPORT	Include Greek (ISO-8859-7) charset support	Disabled
hebrewsupport	_HEBREW_SUPPORT	Include Hebrew (ISO-8859-8) charset support	Disabled
latin5support	_LATIN5_SUPPORT	Include Turkish (Latin 5, ISO-8859-9) charset support	Disabled
latin6support	_LATIN6_SUPPORT	Include Nordic, Latin 6, ISO-8859-10) charset support	Disabled
thaisupport	_THAI_SUPPORT	Include Thai (ISO-8859-11) charset support	Disabled
latin7support	_LATIN7_SUPPORT	Include Latin 7 (ISO-8859-13) charset support	Disabled
latin8support	_LATIN8_SUPPORT	Include Latin 8 (ISO-8859-14) charset support	Disabled
latin9support	_LATIN9_SUPPORT	Include Latin 9 (ISO-8859-15, West Extended) charset support	Disabled
latin10support	_LATIN10_SUPPORT	Include Latin 10 (ISO-8859-16, Romanian) charset support	Disabled
gbsupport	_GB_SUPPORT	Include EUC encoding of GB2312 charset support	Enabled
gbksupport	_GBK_SUPPORT	Include GBK charset support	Enabled
gb18030support	_GB18030_SUPPORT	Include GB18030-0 charset support	Disabled
big5support	_BIG5_SUPPORT	Include BIG5 charset support	Enabled
euckrsupport	_EUCKR_SUPPORT	Include support for EUC encoding of KSC5636 and KSC5601 charsets	Disabled
eucjpsupport	_EUCJP_SUPPORT	Include support for EUC encoding of JISX0201 and JISX0208 charsets	Disabled
shiftjissupport	_SHIFTJIS_SUPPORT	Include support for Shift-JIS encoding of JISX0201 and JISX0208 charsets	Disabled
unicodesupport	_UNICODE_SUPPORT	Include UNICODE (ISO-10646-1 and UTF-8 encoding) support	Enabled
rbfsupport	_RBF_SUPPORT	Include RBFfont support	Enabled
rbf16	_INCORERBF_LATIN1_16	Include incore RBF font of ISO8859-1 8x16 fixed font	Disabled
rbf24	_INCORERBF_LATIN1_24	Include incore RBF font of ISO8859-1 12x24 fixed font	Disabled
rbfgb12	_INCORERBF_GB12	Include incore RBF font of GB2312 12x12 fixed/song font	Enabled
rbfgb16	_INCORERBF_GB16	Include incore RBF font of GB2312 16x16 fixed/song font	Disabled
rbfgb24	_INCORERBF_GB24	Include incore RBF font of GB2312 24x24 fixed/song font	Disabled
rbfgb16hei	_INCORERBF_GB16_H	Include incore RBF font of GB2312 16x16 fixed/hei font	Disabled
rbfbig12	_INCORERBF_BIG5_12	Include incore RBF font of BIG5 12x12 fixed/ming font	Disabled
rbfbig16	_INCORERBF_BIG5_16	Include incore RBF font of BIG5 16x16 fixed/ming font	Disabled
rbfbig24	_INCORERBF_BIG5_24	Include incore RBF font of BIG5 24x24 fixed/ming font	Disabled
rbfkj12	_INCORERBF_KJ12	Include incore RBF font of SHIFT-JIS Kanji fonts 12	Disabled
rbfkj14	_INCORERBF_KJ14	Include incore RBF font of	Disabled

		SHIFT-JIS Kanji fonts 14	
rbfkj16	_INCORERBF_KJ16	include incore RBF font of SHIFT-JIS Kanji fonts 16	Disabled
vbfsupport	_VBF_SUPPORT	Include var bitmap font support	Enabled
fontsserif	_INCOREFONT_SANSSERIF	Include incore VBF font sansserif	Enabled
fontcourier	_INCOREFONT_COURIER	Include incore VBF font courier	Enabled
fontsymbol	_INCOREFONT_SYMBOL	Include incore VBF font symbol	Disabled
fontvgas	_INCOREFONT_VGAS	Include incore VBF font VGAs	Disabled
fonthelv	_INCOREFONT_HELV	Include incore VBF font Helvetica	Disabled
qpfsupport	_QPF_SUPPORT	Include Qt Prerendered Font (QPF) support	Enabled
ft2support	_FT2_SUPPORT	Include FreeType2 Library support	Disabled
ttfsupport	_TTF_SUPPORT	Include TrueType font support	Disabled
ttfcache	_TTF_CACHE_SUPPORT	Include TrueType cache support	Disabled
type1support	_TYPE1_SUPPORT	Include Adobe Type1 font support	Enabled

The options latin2support, latin3support, cyrillicsupport, arabicsupport, greeksupport, hebrewsupport, latin5support, latin6support, thaisupport, latin7support, latin8support, latin9support, latin10support control ISO8859-2 to ISO8859-16 character set support, they are single byte character set. There are supporting for ASCII character and ISO8859-1 (Latin1) build in MiniGUI. No configuration options for these two character sets.

The options gbsupport, gbksupport, gb18030support, big5support, euckrsupport, eucjpsupport, shiftjissupport, unicodesupport control GB2312, GBK, GB18030, BIG5, EUCKR, EUCJP, SHIFTJIS, UNICODE character set/code system support.

The option rbfsupport control whether include the support for Raw Bitmap Font (RBF) font, it is enabled as the default. Because RBF is the default font format, so it is not suggested that user disable the support for this font type.

The options rbfgb12, rbfgb16, rbfgb24 and rbfgb16hei control whether GB2312 12, 16, 24 and 16 bold dot fonts are built in MiniGUI. These configuration options are disabled default and these options are available when option incoreres are enabled.

The options rbfbig12, rbfbig16 and rbfbig24 control whether BIG5 12, 16 and 24-dot font are built in MiniGUI. These configuration options are disabled default and these options are available when option incoreres enabled.

The options rbfkj12, rbfkj14 and rbfkj16 control whether Japanese 12, 14 and 16-dot font are built in MiniGUI. These configuration options are disabled default and these options are available when option incoreres enabled.

The option vbfsupport control whether include support for Variable Bitmap Font (VBF) font, it is enabled default. If this option is disabled, you not only disable the support for VBF font but also disable the VBF font build in MiniGUI. When MiniGUI is running, the runtime option [**`varbitmapfonts`**] section is ignored.

The options fontsserif, fontcourier, fontsymbol, fonthelv, fontvgas control whether support for SanSerif, Courier, Symbol, Helvetica and some VGA font built in MiniGUI. The attention, these fonts are VBF font format, and they do not effect by incoreres option.

The option qpfsupport control whether support for Qt/Embedded Prerendered Font (QPF). Because QPF font uses UNICODE coding, so if support QPF font in MiniGUI, the UNICODE support is enabled automatically. If incoreres option is enabled, some QPF fonts will be built in MiniGUI.

The option type1support controls whether support Adobe Type1 in MiniGUI library. MiniGUI render the Type1 font by libt1. If t1 library is not installed in your system, the configuration will enable this option automatically.

The option ft2support control whether support for FreeType2 library in MiniGUI library. MiniGUI can render the TrueType font by FreeType2 library version 2.3.4. If FreeType2 library is not installed in your system, the configuration will disable this option automatically.

The option ttfsupport control whether support for TrueType in MiniGUI library. MiniGUI also can render the TrueType font by FreeType library version 1.3.0. If FreeType library version 1.3.0 is not installed in your system, the configuration will disable this option automatically. The attention, the interfaces of FreeType 2 are not compatible with FreeType 1.

The option ttfcache control whether support TrueType cache for FreeType1, it is enabled default. If ttfcache need enable, the option ttfsupport should be enabled first.

The option **`--with-mttfcachenum`** uses for appoint the number of the cache block when TrueType cache is enabled. The default value is 10.

The option `--with-ttfcachesize` uses for appoint the size of cache block when TrueType cache is enabled, the default value is 64k.

Table 2.14 and table 2.15 list the TrueType cache related parameters, options and macros.

Table 2.14 TrueType cache related options and macros

Configure option	Macro	Macro value	Memo
--with-mttfcachenum=10	_MAX_TTF_CACHE	10	Default value
--with-mttfcachenum=20		20	
--with-mttfcachenum=40		40	

Table 2.15 TrueType cache related options and macros

Configure option	Macro	Macro value	Memo
--with-ttfcachesize=64	_TTF_CACHE_SIZE	64	Default value
--with-ttfcachesize=128		128	
--with-ttfcachesize=256		256	
--with-ttfcachesize=512		512	
--with-ttfcachesize=1024		1024	

2.2.10 Image File Format Related Options and Macros

MiniGUI support for multiple image file formats, idiographic, MiniGUI include Windows BMP, GIF, JPEG, PNG, PCX, LBM/PBM, TGA and so on. Thereinto, MiniGUI only support Windows BMP in incore resource, so there is not corresponding configuration option; The configuration option of GIF, JPEG, PNG file is enabled; The configuration option of PCX, LBM/PBM, TGA is disabled. It should be noted that if you want to MiniGUI support JECG and PNG picture format, you need to install corresponding libjpeg and libpng libraries into your system, there is the source code of these two function libraries in the MiniGUI-VAR CD-ROM `deplibs/` directory.

The table 2.16 listed image file format related configuration options and macros.

Table 2.16 image file format related configuration options and macros

configuration option	Macro	Comment	Default value
gifsupport	_GIF_FILE_SUPPORT	Support for GIF file	Enable
jpgsupport	_JPG_FILE_SUPPORT	Support for JPG file	Enable
pngsupport	_PNG_FILE_SUPPORT	Support for PNG file	Enable
pcxsupport	_PCX_FILE_SUPPORT	Support for PCX file	Disable
lbmsupport	_LBM_FILE_SUPPORT	Support for LBM/PBM file	Disable
tgasupport	_TGA_FILE_SUPPORT	Support for TGA file	Disable

2.2.11 Appearance Style Related Options and Macros

The window and control have three appearance styles in the MiniGUI: Classic, Flat and Fashion. Table 2.17 listed appearance style configuration options and corresponding macros.

Table 2.17 appearance style related configuration options and macros

configuration option	Macro	Comment	Memo
--with-style=classic	_PC3D_WINDOW_STYLE	Classic style	Default value
--with-style=flat	_GRAY_SCREEN _FLAT_WINDOW_STYLE	This style is fit in with industry control domain, display with gray screen.	
--with-style=fashion	_PHONE_WINDOW_STYLE	Fashion style	
tinyscreen	_TINY_SCREEN	The object board is tiny screen	Disable
fixedbarlen	_USE_FIXED_SB_BARLEN	The scroll bar has certain length	Disable

The table 2.17 also listed two options: tinyscren and fixedbarlen, the tinyscreen tell MiniGUI the differentiate rate of the object board is lower, so MiniGUI is to the best of it's ability and compactly display the content of the system dialog, when MiniGUI display the system dialog. The fixedbarlen tell MiniGUI the scroll bar of the window or control has certain length, it doesn't change along with the size of scroll range.

2.2.12 Control Related Options and Macros

There are two categories controls in the MiniGUI, one is basic control, and the other is extended library control. MiniGUI supports configuration options for all controls. Default, the most of the controls is enabled, but you can disable some controls, so you can reduce the size of the MiniGUI kernel library and extended library.

Except controls, the extskin configuration option can control to include support for skin interface in the MiniGUI extended library.

Table 2.18 give all controls related configuration options and macros.

Table 2.18 control related configuration options and macros

configuration option	Macro	Comment	Default value
ctrlstatic	_CTRL_STATIC	Include STATIC control	Enable
ctrlbutton	_CTRL_BUTTON	Include BUTTON control	Enable
ctrlsimedit	_CTRL_SIMEDIT	Include Simple EDITcontrol	Enable
ctrlsledit	_CTRL_SLEDIT	Include Single-Line EDIT control	Enable
ctrllistbox	_CTRL_LISTBOX	Include LISTBOXcontrol	Enable
ctrlpgbar	_CTRL_PROGRESSBAR	Include PROGRESSBAR control	Enable

ctrltoolbar	_CTRL_TOOLBAR	Include TOOLBAR control	Enable
ctrlnewtoolbar	_CTRL_NEWTOOLBAR	Include NEWTOOLBARcontrol	Enable
ctrlmenubtn	_CTRL_MENUBUTTON	Include MENUBUTTON control	Enable
ctrltrackbar	_CTRL_TRACKBAR	Include TRACKBARcontrol	Enable
ctrlcombobox	_CTRL_COMBOBOX	Include COMBOBOX control	Enable
ctrlpropsheet	_CTRL_PROPSHEET	Include PROPSHEET control	Enable
ctrlscrollview	_CTRL_SCROLLVIEW	Include SCROLLVIEW and SCROLLWINDOW control	Enable
ctrltextedit	_CTRL_TEXTEDIT	Include base on SCROLLVIEW TEXTEDIT control	Enable
extctrlmonthcal	_EXT_CTRL_MONTHCAL	Include MONTHCALENDAR control in the MiniGUIExt library	Enable
extctrltreeview	_EXT_CTRL_TREEVIEW	Include TREEVIEW control in the MiniGUIExt library	Enable
extctrlspinbox	_EXT_CTRL_SPINBOX	Include SPINBOX control in the MiniGUIExt library	Enable
extctrlcoolbar	_EXT_CTRL_COOLBAR	Include COOLBAR control in the MiniGUIExt library	Enable
extctrllistview	_EXT_CTRL_LISTVIEW	Include LISTVIEW control in the MiniGUIExt library	Enable
extctrliconview	_EXT_CTRL_ICONVIEW	Include ICONVIEW control in the MiniGUIExt library	Enable
extctrlgridview	_EXT_CTRL_GRIDVIEW	Include GRIDVIEW control in the MiniGUIExt library	Disable
extctrlanimation	_EXT_CTRL_ANIMATION	Include ANIMATION control and support for GIF89a file in the MiniGUIExt library	Enable
extskin	_EXT_SKIN	Include support for skin in the MiniGUIExt library	Enable

2.2.13 Other Options and Macros

MiniGUI implemented some function families of the standard C function libraries to be fit in with all kinds of embedded operating system environment, it include malloc function family (malloc, calloc, free function and so on), stdio format input and output function family (printf, sprintf and so on) and POSIX thread function library interface (pthread_create, sem_post and so on). Default, these function families compile configuration options is disabled, and that they are useful in the some traditional embedded operating system based on thread and task. If you want to enable these options in the some operating systems, you can refer to 2.2.1 chapter. Table 2.19 listed MiniGUI implemented C library interface configuration options and corresponding macros.

Table 2.19 MiniGUI implemented C library interface related configurations and macros

configuration option	Macro	Comment	Default value
ownmalloc	_USE_OWN_MALLOC	Use MiniGUI implemented malloc function family	Disable
ownstdio	_USE_OWN_STDIO	Use MiniGUI implemented stdio format input and output function family	Disable

ownpthread	_USE_OWN_PTHREAD	Use MiniGUI implemented thread function family	Disable

Otherwise, you must define two macros: __MINIGUI_LIB__ and __MGEXT_LIB__, when you use yourself makefile to compile MiniGUI function library in the Non-GNU development environment.

Table 2.20 other compile macros

Macro	Comment	Memo
__MINIGUI_LIB__	Compile MiniGUI library macro	You must define these two macros, when you use the Non-GNU makefile
__MGEXT_LIB__	Compile MiniGUI extended library macro	

2.3Minimum Configuration Options

In this chapter, we will give an example of minimum configuration options in MiniGUI.

2.3.1 Using GNU Configure Script

There is a buildlib-min script in the MiniGUI source codes build directory. The buildlib-min script will be as the following:

```
#!/bin/sh

./configure \
    --disable-micemoveable \
    --disable-dblclk \
    --disable-cursor \
    --disable-mousecalibrate \
    --disable-clipboard \
    --disable-vbfsupport \
    --disable-rbfgb12 \
    --disable-fontsserif \
    --disable-fontcourier \
    --disable-qpfsupport \
    --disable-ttfsupport \
    --disable-latin9support \
    --disable-gbsupport \
    --disable-gbksupport \
    --disable-big5support \
    --disable-unicodesupport \
    --disable-savebitmap \
    --disable-jpgsupport \
    --disable-pngsupport \
    --disable-gifsupport \
    --disable-imegb2312 \
    --disable-imegb2312py \
    --disable-aboutdlg \
    --disable-savescreen \
    --disable-adv2dapi \
    --disable-videoqvfb \
    --disable-qvfbial \
    --disable-extctrlmonthcal \
    --disable-extctrltreeview \
    --disable-extctrlspinbox \
    --disable-extctrlcoolbar \
    --disable-extctrllistview \
    --disable-extctrliconview \
    --disable-extctrlgridview \
    --disable-extctrlanimation \
    --disable-extskin \
    --disable-libvcongui \
    --disable-nativeps2 \
    --disable-nativeimps2 \
    --disable-nativems \
    --disable-nativems3 \
    --disable-nativegpm \
    --with-style=flat
```

By this script, you can configure MiniGUI to the minimum function library that only supports ISO8859-1 charset.

- Compiling MiniGUI to be MiniGUI-Threads.
- No support for moving window by using mouse.
- No support for double click mouse button.
- No support for cursor.
- No support for code doing mouse calibration.
- No support for clipboard.
- No support for VBF font.
- No including RBF GB2312 12x12 fixed/song incoreres font.
- No including sansserif incoreres font.
- No including courier incoreres font.
- No support for Qt Prerendered Font(QPF).
- No support for TrueType font.
- No support for Latin 9(ISO-8859-15, West Extended) charset.
- No support for EUC GB2312 charset.
- No support for GBK charset.
- No support for BIG5 charset.
- No support for UNICODE (ISO-10646-1and UTF-8).
- No support for BITMAP saving function.
- No support for JPG image format.
- No support for PNG image format.
- No support for GIF image format.
- No support for GB2312 input method.
- No support for GB2312 intelligent pinyin input method.
- No including “About MiniGUI” dialog box.
- No support for screen save function.
- No support for advanced 2D graphics APIs
- No support for Qt Virtual FrameBuffer NEWGAL engine.

- No building QVFB IAL engine.
- No including MONTHCALENDAR control in MiniGUIExt library.
- No including TREEVIEW control in MiniGUIExt library.
- No including SPINBOX control in MiniGUIExt library.
- No including COOLBAR control in MiniGUIExt library.
- No including LISTVIEW control in MiniGUIExt library.
- No including ICONVIEW control in MiniGUIExt library.
- No including GRIDVIEW control in MiniGUIExt library.
- No including skin support in MiniGUIExt library.
- No including ANIMATION control in MiniGUIExt library, no support for GIF87a/GIF89a.
- No support for Virtual Console on Linux.
- No building the native engine subdriver for PS2 mouse.
- No building the native engine subdriver for IntelligentMouse (IMPS/2).
- No building the native engine subdriver for old MS serial mouse.
- No building the native engine subdriver for MS3 mouse.
- No building the native engine subdriver for GPM daemon.
- The style of MiniGUI is flat.

Based on the configuration above, you can also delete some functions if you want. For example, if you do not use menu button control in your application, you can add **`--disable-ctrlmenubtn`** option in the configuration script above, so there is not menu button control in your compiled functions library, the MiniGUI functions library is made smaller.

2.3.2 Corresponding mgconfig.h

The **`mgconfig.h`** file to be generated in the configuration script above, listed as follows:

```
/* mgconfig.h.  Generated by configure.  */
/* mgconfig.h.in.  Generated from configure.in by autoheader.  */

/* Define to one of `_getb67', `GETB67', `getb67' for Cray-2 and Cray-YMP
   systems. This function is required for `alloca.c' support on those systems.
   */
/* #undef CRAY_STACKSEG_END */

/* Define to 1 if using `alloca.c'. */
/* #undef C_ALLOCA */

/* Define to 1 if you have `alloca', as a function or macro. */
#define HAVE_ALLOCA 1

/* Define to 1 if you have <alloca.h> and it should be used (not on Ultrix).
   */
```

```
#define HAVE_ALLOCA_H 1

/* Define to 1 if you have the <dirent.h> header file, and it defines `DIR'.
   */
#define HAVE_DIRENT_H 1

/* Define to 1 if you have the <dlfcn.h> header file. */
#define HAVE_DLFCN_H 1

/* Define to 1 if you don't have `vprintf' but do have `_doprnt.' */
/* #undef HAVE_DOPRNT */

/* Define to 1 if you have the `getpagesize' function. */
#define HAVE_GETPAGESIZE 1

/* Define to 1 if you have the `getpt' function. */
#define HAVE_GETPT 1

/* Define to 1 if you have the <inttypes.h> header file. */
#define HAVE_INTTYPES_H 1

/* Define to 1 if you have the <locale.h> header file. */
#define HAVE_LOCALE_H 1

/* Define to 1 if you have the <math.h> header file. */
#define HAVE_MATH_H 1

/* Define to 1 if you have the <memory.h> header file. */
#define HAVE_MEMORY_H 1

/* Define to 1 if you have a working `mmap' system call. */
#define HAVE_MMAP 1

/* Define to 1 if you have the <ndir.h> header file, and it defines `DIR'. */
/* #undef HAVE_NDIR_H */

/* Define to 1 if you have the `setenv' function. */
#define HAVE_SETENV 1

/* Define to 1 if you have the `setlocale' function. */
#define HAVE_SETLOCALE 1

/* Define to 1 if you have the <stdint.h> header file. */
#define HAVE_STDINT_H 1

/* Define to 1 if you have the <stdlib.h> header file. */
#define HAVE_STDLIB_H 1

/* Define to 1 if you have the `strcasecmp' function. */
#define HAVE_STRCASECMP 1

/* Define to 1 if you have the `strdup' function. */
#define HAVE_STRDUP 1

/* Define to 1 if you have the `strerror' function. */
#define HAVE_STRERROR 1

/* Define to 1 if you have the <strings.h> header file. */
#define HAVE_STRINGS_H 1

/* Define to 1 if you have the <string.h> header file. */
#define HAVE_STRING_H 1

/* Define to 1 if you have the <sys/dir.h> header file, and it defines `DIR'.
   */
/* #undef HAVE_SYS_DIR_H */

/* Define to 1 if you have the <sys/ndir.h> header file, and it defines `DIR'.
   */
/* #undef HAVE_SYS_NDIR_H */

/* Define to 1 if you have the <sys/stat.h> header file. */
#define HAVE_SYS_STAT_H 1

/* Define to 1 if you have the <sys/time.h> header file. */
```

```
#define HAVE_SYS_TIME_H 1

/* Define to 1 if you have the <sys/types.h> header file. */
#define HAVE_SYS_TYPES_H 1

/* Define to 1 if you have the <termio.h> header file. */
#define HAVE_TERMIO_H 1

/* Define to 1 if you have the `tmpfile' function. */
#define HAVE_TMPFILE 1

/* Define to 1 if you have the <unistd.h> header file. */
#define HAVE_UNISTD_H 1

/* Define to 1 if you have the `vprintf' function. */
#define HAVE_VPRINTF 1

/* Binary age of MiniGUI */
#define MINIGUI_BINARY_AGE 0

/* Interface age of MiniGUI */
#define MINIGUI_INTERFACE_AGE 0

/* Major version of MiniGUI */
#define MINIGUI_MAJOR_VERSION 2

/* Micro version of MiniGUI */
#define MINIGUI_MICRO_VERSION 3

/* Minor version of MiniGUI */
#define MINIGUI_MINOR_VERSION 0

/* Define to the address where bug reports for this package should be sent. */
#define PACKAGE_BUGREPORT ""

/* Define to the full name of this package. */
#define PACKAGE_NAME ""

/* Define to the full name and version of this package. */
#define PACKAGE_STRING ""

/* Define to the one symbol short name of this package. */
#define PACKAGE_TARNAME ""

/* Define to the version of this package. */
#define PACKAGE_VERSION ""

/* If using the C implementation of alloca, define if you know the
   direction of stack growth for your system; otherwise it will be
   automatically deduced at run-time.
        STACK_DIRECTION > 0 => grows toward higher addresses
        STACK_DIRECTION < 0 => grows toward lower addresses
        STACK_DIRECTION = 0 => direction of growth unknown */
/* #undef STACK_DIRECTION */

/* Define to 1 if you have the ANSI C header files. */
#define STDC_HEADERS 1

/* Define to 1 if you can safely include both <sys/time.h> and <time.h>. */
#define TIME_WITH_SYS_TIME 1

/* Define to 1 if your <sys/time.h> declares `struct tm'. */
/* #undef TM_IN_SYS_TIME */

/* Define if compile for Win32 platform */
/* #undef WIN32 */

/* Define if include IAL engine for ABSSIG */
/* #undef _ABSSIG_IAL */

/* Define if include IAL engine for Arca Tpanel Ads7846 */
/* #undef _ADS7846_IAL */

/* Define if include IAL engine for ADS */
/* #undef _ADS_IAL */
/* Define if include advanced 2D graphics APIs */
/* #undef _ADV_2DAPI */

/* Define if support Arabic charset */
/* #undef _ARABIC_SUPPORT */

/* Define if include IAL engine for ARM3000 */
/* #undef _ARM3000_IAL */

/* Define if include the automatic IAL engine */
/* #undef _AUTO_IAL */

/* Define if support BIG5 charset */
/* #undef _BIG5_SUPPORT */

/* Define if include clipboard support */
/* #undef _CLIPBOARD_SUPPORT */

/* Define if include the common IAL engine */
/* #undef _COMM_IAL */

/* Define if use coordinate transformation */
/* #undef _COOR_TRANS */

/* Define if include BUTTON control */
#define _CTRL_BUTTON 1

/* Define if include COMBOBOX control */
#define _CTRL_COMBOBOX 1

/* Define if include LISTBOX control */
#define _CTRL_LISTBOX 1

/* Define if include MENUBUTTON control */
#define _CTRL_MENUBUTTON 1

/* Define if include NEWTOOLBAR control */
#define _CTRL_NEWTOOLBAR 1

/* Define if include PROGRESSBAR control */
#define _CTRL_PROGRESSBAR 1

/* Define if include PROPSHEET control */
#define _CTRL_PROPSHEET 1

/* Define if include SCROLLVIEW control */
#define _CTRL_SCROLLVIEW 1

/* Define if include SIMEDIT control */
#define _CTRL_SIMEDIT 1

/* Define if include SLEDIT control */
#define _CTRL_SLEDIT 1

/* Define if include STATIC control */
#define _CTRL_STATIC 1

/* Define if include TEXTEDIT control */
#define _CTRL_TEXTEDIT 1

/* Define if include TOOLBAR control */
#define _CTRL_TOOLBAR 1

/* Define if include TRACKBAR control */
#define _CTRL_TRACKBAR 1

/* Define if include cursor support */
/* #undef _CURSOR_SUPPORT */

/* Define if support Cyrillic charset */
/* #undef _CYRILLIC_SUPPORT */

/* Define if build with debugging messages */
/* #undef _DEBUG */
/* Define if include IAL engine for DirectFB */
/* #undef _DFB_IAL */

/* Define if include IAL engine for DM270 */
/* #undef _DM270_IAL */

/* Define if include IAL engine for Intel DMG STB Remote Controller */
/* #undef _DMGSTB_IAL */

/* Define if mouse button can do double click */
/* #undef _DOUBLE_CLICK */

/* Define if include the dummy IAL engine */
#define _DUMMY_IAL 1

/* Define if include the em85 IAL engine */
/* #undef _EM85_IAL */

/* Define if include the em86 IAL engine */
/* #undef _EM86_IAL */

/* Define if include IAL engine for EMBEST ARM2410 */
/* #undef _EMBEST2410_IAL */

/* Define if include IAL engine for EMBEST ARM44B0 */
/* #undef _EMBEST44B0_IAL */

/* Define if include IAL engine for EP7211 */
/* #undef _EP7211_IAL */

/* Define if support EUCJP charset */
/* #undef _EUCJP_SUPPORT */

/* Define if support EUCKR charset */
/* #undef _EUCKR_SUPPORT */

/* Define if include IAL engine for xScale EVMV 1.0 */
/* #undef _EVMV10_IAL */

/* Define if include ANIMATION control */
/* #undef _EXT_CTRL_ANIMATION */

/* Define if include COOLBAR control */
/* #undef _EXT_CTRL_COOLBAR */

/* Define if include GRIDVIEW control */
/* #undef _EXT_CTRL_GRIDVIEW */

/* Define if include ICONVIEW control */
/* #undef _EXT_CTRL_ICONVIEW */

/* Define if include LISTVIEW control */
/* #undef _EXT_CTRL_LISTVIEW */

/* Define if include MONTHCALENDAR control */
/* #undef _EXT_CTRL_MONTHCAL */

/* Define if include SPINBOX control */
/* #undef _EXT_CTRL_SPINBOX */

/* Define if include TREEVIEW control */
/* #undef _EXT_CTRL_TREEVIEW */

/* Define if include skin support */
/* #undef _EXT_SKIN */

/* Define if include IAL engine for FFT7202 */
/* #undef _FFT7202_IAL */

/* Define if include IAL engine for EM85xx Front Panel and Remote Controller
   */
/* #undef _FIP_IAL */

/* Define if include fixed math routines */
```

```
#define _FIXED_MATH 1

/* Define if window style is flat */
#define _FLAT_WINDOW_STYLE 1

/* Define if include IAL engine for FXRM9200 */
/* #undef _FXRM9200_IAL */

/* Define if support GB18030 charset */
/* #undef _GB18030_SUPPORT */

/* Define if support GBK charset */
/* #undef _GBK_SUPPORT */

/* Define if support GB2312 charset */
/* #undef _GB_SUPPORT */

/* Define if support GIF bmp file format */
/* #undef _GIF_FILE_SUPPORT */

/* Define if include GPM mouse subdriver */
/* #undef _GPM_SUPPORT */

/* Define if is gray screen */
#define _GRAY_SCREEN 1

/* Define if support Greek charset */
/* #undef _GREEK_SUPPORT */

/* Define if have math library */
#define _HAVE_MATH_LIB 1

/* Define if your Linux have text mode */
#define _HAVE_TEXT_MODE 1

/* Define if support Hebrew charset */
/* #undef _HEBREW_SUPPORT */

/* Define if include IAL engine for Helio Touch Panel */
/* #undef _HELIO_IAL */

/* Define if include IAL engine for HHARM2410R3 touch screen */
/* #undef _HH2410R3_IAL */

/* Define if include IAL engine for HHARM2440 touch screen */
/* #undef _HH2440_IAL */

/* Define if include IAL engine for HHARM44B0 touch screen */
/* #undef _HH44B0_IAL */

/* Define if include the IAL engine for HHCF5249 PS/2 keyboard and IrDA */
/* #undef _HH5249KBDIR_IAL */

/* Define if include GB2312 IME */
/* #undef _IME_GB2312 */

/* Define if include GB2312 Intelligent Pinyin IME module */
/* #undef _IME_GB2312_PINYIN */

/* Define if include IMPS2 mouse subdriver */
/* #undef _IMPS2_SUPPORT */

/* Define if include in-core font: Courier */
/* #undef _INCOREFONT_COURIER */

/* Define if include in-core font: Helvetica */
/* #undef _INCOREFONT_HELV */

/* Define if include in-core font: SansSerif */
/* #undef _INCOREFONT_SANSSERIF */

/* Define if include in-core font: Symbol */
/* #undef _INCOREFONT_SYMBOL */

/* Define if include in-core font: VGAS */
/* #undef _INCOREFONT_VGAS */

/* Define if include in-core font: 12x12 fixed/ming RBF for BIG5 */
/* #undef _INCORERBF_BIG5_12 */

/* Define if include in-core font: 16x16 fixed/ming RBF for BIG5 */
/* #undef _INCORERBF_BIG5_16 */

/* Define if include in-core font: 24x24 fixed/ming RBF for BIG5 */
/* #undef _INCORERBF_BIG5_24 */

/* Define if include in-core font: 12x12 fixed/song RBF for GB2312 */
/* #undef _INCORERBF_GB12 */

/* Define if include in-core font: 16x16 fixed/song RBF for GB2312 */
/* #undef _INCORERBF_GB16 */

/* Define if include in-core font: 16x16 fixed/hei RBF for GB2312 */
/* #undef _INCORERBF_GB16_HEI */

/* Define if include in-core font: 24x24 fixed/song RBF for GB2312 */
/* #undef _INCORERBF_GB24 */

/* Define if include in-core font: 12-Pixel Kanji RBFs for SHIFT-JIS */
/* #undef _INCORERBF_KJ12 */

/* Define if include in-core font: 14-Pixel Kanji RBFs for SHIFT-JIS */
/* #undef _INCORERBF_KJ14 */

/* Define if include in-core font: 16-Pixel Kanji RBFs for SHIFT-JIS */
/* #undef _INCORERBF_KJ16 */

/* Define if include in-core font: 8x16 fixed RBF for ISO8859-1 */
/* #undef _INCORERBF_LATIN1_16 */

/* Define if include in-core font: 12x24 fixed RBF for ISO8859-1 */
/* #undef _INCORERBF_LATIN1_24 */

/* Define if build MiniGUI for no file I/O system */
/* #undef _INCORE_RES */

/* Define if include IAL engine for iPAQ */
/* #undef _IPAQ_IAL */

/* Define if support JPEG bmp file format */
/* #undef _JPG_FILE_SUPPORT */

/* Define if use the German keyboard layout */
/* #undef _KBD_LAYOUT_DE */

/* Define if use the German-Latin1 keyboard layout */
/* #undef _KBD_LAYOUT_DELATIN1 */

/* Define if use the Spanish keyboard layout */
/* #undef _KBD_LAYOUT_ES */

/* Define if use the Spanish CP850 keyboard layout */
/* #undef _KBD_LAYOUT_ESCP850 */

/* Define if use the French keyboard layout */
/* #undef _KBD_LAYOUT_FR */

/* Define if use the French PC keyboard layout */
/* #undef _KBD_LAYOUT_FRPC */

/* Define if use the Italian keyboard layout */
/* #undef _KBD_LAYOUT_IT */

/* Define if include IAL engine for L7200 */
/* #undef _L7200_IAL */

/* Define if support Latin 10 charset */
/* #undef _LATIN10_SUPPORT */

/* Define if support Latin 2 charset */
/* #undef _LATIN2_SUPPORT */

/* Define if support Latin 3 charset */
/* #undef _LATIN3_SUPPORT */

/* Define if support Latin 4 charset */
/* #undef _LATIN4_SUPPORT */

/* Define if support Latin 5 charset */
/* #undef _LATIN5_SUPPORT */

/* Define if support Latin 6 charset */
/* #undef _LATIN6_SUPPORT */

/* Define if support Latin 7 charset */
/* #undef _LATIN7_SUPPORT */

/* Define if support Latin 8 charset */
/* #undef _LATIN8_SUPPORT */

/* Define if support Latin 9 charset */
/* #undef _LATIN9_SUPPORT */

/* Define if support LBM bmp file format */
/* #undef _LBM_FILE_SUPPORT */

/* Define if include vcongui support */
/* #undef _LIB_VCONGUI */

/* Define if build MiniGUI-Processes */
/* #undef _LITE_VERSION */

/* Define if compile max ttf cahce number for 10 (default value) */
/* #undef _MAX_TTF_CACHE */

/* Define if include IAL engine for uClinux touch screen palm/mc68ez328 */
/* #undef _MC68X328_IAL */

/* Define if build MiniGUI-Processes */
/* #undef _MGRM_PROCESSES */

/* Define if build MiniGUI-Standalone */
/* #undef _MGRM_STANDALONE */

/* Define if build MiniGUI-Threads */
#define _MGRM_THREADS 1

/* Define if include About MiniGUI Dialog Box */
/* #undef _MISC_ABOUTDLG */

/* Define if include code for mouse calibration */
/* #undef _MISC_MOUSECALIBRATE */

/* Define if include code for screenshots */
/* #undef _MISC_SAVESCREEN */

/* Define if we can move window by mouse */
/* #undef _MOVE_WINDOW_BY_MOUSE */

/* Define if include IAL engine for MPC823 */
/* #undef _MPC823_IAL */

/* Define if include MS3 mouse subdriver */
/* #undef _MS3_SUPPORT */

/* Define if include messages' string names */
/* #undef _MSG_STRING */

/* Define if include MS mouse subdriver */
/* #undef _MS_SUPPORT */

/* Define if include native (Linux console) IAL engine */
#define _NATIVE_IAL_ENGINE 1

/* Define if include NEWGAL engine for BF533 OSD via SPI */
/* #undef _NEWGAL_ENGINE_BF533 */

/* Define if include NEWGAL engine for Common LCD */
/* #undef _NEWGAL_ENGINE_COMMLCD */

/* Define if include NEWGAL engine for DirectFB */
/* #undef _NEWGAL_ENGINE_DFB */

/* Define if include dummy NEWGAL engine */
#define _NEWGAL_ENGINE_DUMMY 1

/* Define if include NEWGAL engine for EM85xx OSD */
/* #undef _NEWGAL_ENGINE_EM85XXOSD */

/* Define if include FrameBuffer console NEWGAL engine */
#define _NEWGAL_ENGINE_FBCON 1

/* Define if include NEWGAL engine for mb93493 YUV FrameBuffer driver */
/* #undef _NEWGAL_ENGINE_MB93493 */

/* Define if include Qt Virtual FrameBuffer NEWGAL engine */
/* #undef _NEWGAL_ENGINE_QVFB */

/* Define if include Shadow NEWGAL engine */
/* #undef _NEWGAL_ENGINE_SHADOW */

/* Define if include NEWGAL engine for SVPXX OSD */
/* #undef _NEWGAL_ENGINE_SVPXXOSD */

/* Define if include NEWGAL engine for UTPMC */
/* #undef _NEWGAL_ENGINE_UTPMC */

/* Define if include windows Virtual FrameBuffer NEWGAL engine */
/* #undef _NEWGAL_ENGINE_WVFB */

/* Define if include IAL engine for OKWAP Palm2 */
/* #undef _PALMII_IAL */

/* Define if window style is classic */
/* #undef _PC3D_WINDOW_STYLE */

/* Define if support PCX bmp file format */
/* #undef _PCX_FILE_SUPPORT */

/* Define if window style is fashion */
/* #undef _PHONE_WINDOW_STYLE */

/* Define if support PNG bmp file format */
/* #undef _PNG_FILE_SUPPORT */

/* Define if include PS2 mouse subdriver */
/* #undef _PS2_SUPPORT */

/* Define if include IAL engine for PX255B */
/* #undef _PX255B_IAL */

/* Define if support QPF font */
/* #undef _QPF_SUPPORT */

/* Define if include the QVFB IAL engine */
/* #undef _QVFB_IAL */

/* Define if include the random IAL engine */
/* #undef _RANDOM_IAL */

/* Define if support raw bitmap fonts */
#define _RBF_SUPPORT 1

/* Define to 0 if use clockwise rotation of screen, otherwise to 1 */
/* #undef _ROT_DIR_CW */

/* Define if include SaveBitmap function */
/* #undef _SAVE_BITMAP */

/* Define if support SHIFTJIS charset */
/* #undef _SHIFTJIS_SUPPORT */

/* Define if include IAL engine for touch screen of SkyEye EP7312 simulation
   */
/* #undef _SKYEYE_EP7312_IAL */

/* Define if include IAL engine for SMDK2410 touch screen */
/* #undef _SMDK2410_IAL */

/* Define if build MiniGUI-Standalone */
/* #undef _STAND_ALONE */

/* Define if include IAL engine for WinBond SPVXX */
/* #undef _SVPXX_IAL */

/* Define if include IAL engine for MT T800 */
/* #undef _T800_IAL */

/* Define if include IAL engine for Tongfang STB */
/* #undef _TFSTB_IAL */

/* Define if support TGA bmp file format */
/* #undef _TGA_FILE_SUPPORT */

/* Define if support Thai charset */
/* #undef _THAI_SUPPORT */

/* Define if include IAL engine for THOR */
/* #undef _THOR_IAL */

/* Define if the unit of timer is 10ms */
#define _TIMER_UNIT_10MS 1

/* Define if it is tiny screen */
/* #undef _TINY_SCREEN */

/* Define if trace message dispatching of MiniGUI */
/* #undef _TRACE_MSG */

/* Define if compile max ttf cahce size for 64k in KB */
/* #undef _TTF_CACHE_SIZE */

/* Define if include ttf cache */
/* #undef _TTF_CACHE_SUPPORT */

/* Define if support TrueType font based on FreeType 1.3 */
/* #undef _TTF_SUPPORT */

/* Define if support Adobe Type1 fonts */
/* #undef _TYPE1_SUPPORT */

/* Define if include IAL engine for UCB1X00 */
/* #undef _UCB1X00_IAL */

/* Define if support UNICODE */
/* #undef _UNICODE_SUPPORT */

/* Define if use fixed scrollbar bar length */
/* #undef _USE_FIXED_SB_BARLEN */

/* Define if use minigui_entry function in MiniGUI */
/* #undef _USE_MINIGUIENTRY */

/* Define if use new GAL interfaces */
#define _USE_NEWGAL 1

/* Define if use own implementation of malloc functions */
/* #undef _USE_OWN_MALLOC */

/* Define if use own implementation of pthread functions */
/* #undef _USE_OWN_PTHREAD */

/* Define if use own implementation of stdio functions */
/* #undef _USE_OWN_STDIO */
/* Define if include the UTPMC IAL engine */
/* #undef _UTPMC_IAL */

/* Define if support var bitmap fonts */
/* #undef _VBF_SUPPORT */

/* Define if include IAL engine for NEC VR4181 */
/* #undef _VR4181_IAL */

/* Define if include the WVFB IAL engine */
/* #undef _WVFB_IAL */

/* Define if compile for Cygwin platform */
/* #undef __CYGWIN__ */

/* Define if compile for OpenDarwin */
/* #undef __DARWIN__ */

/* Define if compile for eCos */
/* #undef __ECOS__ */

/* Define if compile for Linux */
#define __LINUX__ 1

/* Define if compile for non-UNIX like OS */
/* #undef __NOUNIX__ */

/* Define if compile for Nucleus */
/* #undef __NUCLEUS__ */

/* Define for Linux FrameBuffer console (used by Shadow NEGAL engine) */
/* #undef __TARGET_FBCON__ */

/* Define for Linux QVFB (used by Shadow NEGAL engine) */
/* #undef __TARGET_QVFB__ */

/* Define for Philips STB810 target */
/* #undef __TARGET_STB810__ */

/* Define for unknown target */
#define __TARGET_UNKNOWN__ 1

/* Define for VirualFone ANVIL target */
/* #undef __TARGET_VFANVIL__ */

/* Define for VxWorks on i386 */
/* #undef __TARGET_VXI386__ */

/* Define for the Windows QVFB (used by Shadow NEGAL engine) */
/* #undef __TARGET_WVFB__ */

/* Define if compile for ThreadX */
/* #undef __THREADX__ */

/* Define if compile for uC/OS-II */
/* #undef __UCOSII__ */

/* Define if compile for VxWorks */
/* #undef __VXWORKS__ */

/* Define if compile for Winbond SWLinux */
/* #undef __WINBOND_SWLINUX__ */

/* Define if compile for uClinux */
/* #undef __uClinux__ */

/* Define to empty if `const' does not conform to ANSI C. */
/* #undef const */

/* Define as `__inline' if that's what the C compiler calls it, or to nothing
   if it is not supported. */
/* #undef inline */

/* Define to `unsigned' if <sys/types.h> does not define. */
/* #undef size_t */
```

2.4 Compiling and Installing MiniGUI

2.4.1 Compiling and Installing MiniGUI in the GNU Development Environment

If you configure MiniGUI with configure script in GNU development environment, you can compile and install MiniGUI with make tool.

For example, assuming that you used MiniGUI-VAR for Linux product, in the PC computer for running Linux, you can execute several commands as the following in your MiniGUI source code directory to configure, compile and install MiniGUI to your system.

```
user$ ./configure
user$ make
user$ su -c ‘make install’
```

You can also use configure script to specify a cross-compiling directory and installing directory and so on.

2.4.2 Compiling and Installing MiniGUI in the Non-GNU Development Environment

In the Non-GNU development environment (generally, it is Windows platform), we first organize MiniGUI source code solution for project of special Integration Development Environment (for example, Tornado and ADS). Secondly, we compile MiniGUI. At last, we compile MiniGUI application.

But using cygwin development environment for Windows platform, it is very convenient. We can compile and install MiniGUI. In theory, this method is applicable to any development environment, which runs on Windows platform, so we will give detailed description on this method in this chapter.

Cygwin is an open source software project and Linux-like environment for Windows. After installing cygwin on Windows, we can execute many applications of Linux platform, for example, BASH script, VIM editor, PERL script interpreter, make tool of Linux, gcc compiler and so on. In the cygwin environment, we can also call other Windows applications. Thus, if we write makefile for MiniGUI according to GNU rules and use make tool of cygwin to call corresponding compiler and linker, we can compile and generate MiniGUI functions library.

Many OSes (Operating System) development environments include cygwin such as OSE. If there is not cygwin in your development environment, you can download and install it from http://www.cygwin.com. Please make sure you have installed make tool, compiler and BASH shell script software package and so on.

In MiniGUI source code, in order to compile MiniGUI conveniently in the Non-GNU development environment, the following things have been done.

- In order to distinguish makefile of cygwin from GNU makefile, the GNU makefile is generated by configure tool, the makefile of cygwin has **.ng** suffix (the **.ng** expresses non-GNU).
- Provide template header file for special platform and operating system, the rules[1] of nomenclature is like config-<os>-<platform>.h.
- Provide a self-compiled rule file (the name is **rules.make**). The **rules.make** is in the MiniGUI source code top directory. In rules.make, we need provide different TARGET_RULES value for different OS development environment.
- Provide some spare **rules.make** files for different OS (Operating System) development environment. We save these files to the MiniGUI source code **build/** directory. The rules of nomenclature in these files is like **rules-<platform>.<os>.**

Firstly, we copy build/ config-<os>-<platform>.h to MiniGUI source code top directory, and rename it as mgconfig.h. Secondly we modify **rules.make** file according to actual development environment. Lastly, we compile MiniGUI using cygwin make command. For example, we want to compile MiniGUI for VxWorks X86 platform (rules file corresponding with **build/rules-pc.vxworks**[3]), we need follow the following step:

Copy build/config-vxworks-i386.h to MiniGUI source code top directory, and rename it as mgconfig.h (we resume that current directory is MiniGUI source code top directory):

```
cygwin$ cp build/config-vxworks-i386.h  mgconfig.h
```

Modify TARGET_RULES value in rules.make file:

[3] Note that we only provide this file in the VxWorks OS MiniGUI-VAR product.

```
TARGET_RULES=build/rules-pc.vxworks
```

Then we compile MiniGUI using make tool of cygwin:

```
cygwin$ /usr/bin/make -f makefile.ng
```

Note that **`makefile.ng`** supports commands of clean and make. If you execute the command as follow:

```
cygwin$ /usr/bin/make -f makefile.ng install
```

You can install MiniGUI header files and library to the directory, which is specified by **`rules-<platform>.<os>`**. If you execute the command as the following:

```
cygwin$ /usr/bin/make -f makefile.ng clean
```

You can clean all object files to compile afresh.

Note: if you modify **`mgconfig.h`** and other files in the cygwin environment, first of all you execute the command above to clean all object files, then compile MiniGUI afresh.

By using cygwin environment and **`makefile.ng`** to compile MiniGUI, our main work is in editing right **`rules.make`** file, actually. You must define variables accurately in the table 2.21, when you compile **`rules.make`** under yourself development environment.

Table 2.21 the variables needed by makefile.ng

Variants name	Purpose	Memo
CC	Specify C compiler	
CPP	Specify C++ compiler	
AR	Specify archiving tool, the tool is used to generate static library	
RANLIB	Specify static library index tool	
MAKE	Specify make tool	Generally, the make tool is /usr/bin/make in the cygwin environment
ARFLAGS	The option that controls the archiving tool generate static library	
COFLAG	The option that it control the compiler to compile, but not link	
OBJ	The suffix name of the object file	
LIBA	The suffix of the static library file	
PREFIX	The prefix of the installation directory	
INCS	Specify the search directory option of head file	
CFLAGS	The C compiler option	

`build/rules-pc.vxworks` file was listed as follows:

```
# rules for pc-vxworks

AS=
CC=ccpentium
CXX=c++pentium
CPP=ccpentium
AR=arpentium
RANLIB=ranlibpentium
MAKE=/usr/bin/make

ARFLAGS=crus
COFLAG=-c

OBJ=o
LIBA=a

PREFIX=c:/cross

#vxworks
TARGET_DIR=C:/Tornado2.2x86/target

INCS+=-I${TARGET_DIR}/h

CFLAGS+=-g -mcpu=pentium  -march=pentium -Wall -DTOOL_FAMILY=gnu -DTOOL=gnu -D_WRS_KERNEL -DCPU=
PENTIUM
```

Note that the make tool will install MiniGUI header files to the **$PREFIX/include/minigui** directory under the **makefile.ng** project file of cygwin, the function libraries were installed to the **$PREFIX/lib/** directory. The **rules.make** file above will install MiniGUI header files to the **c:/cross/include/minigui** directory and MiniGUI libraries to the **c:/cross/lib** directory.

Referring to table 2.21 and the **rules.make** file above, you can write correct rules.make file based on actually development environment.

Because the format of the **makefile.ng** is compatible with GNU makefile, so we can use **makefile.ng** to compile MiniGUI in the Linux environment, actually. This kind of circumstance usually occurs during using cross-compile tool chain for uClinux. If you work in the Linux environment, you can execute make command.

```
user$ make -f makefile.ng
```

About other contents related with portion and configuration of MiniGUI, please refer to Chapter 18 "*GAL and IAL Engines*" and Appendix A "*A Universal Startup API for RTOSes*" in MiniGUI Programming Guide V2.0.4/1.6.10.

3 MiniGUI runtime configuration options

In this chapter, we describe the MiniGUI runtime configuration options, which effect some actions about MiniGUI running, for example, running GAL and IAL used, device font, bitmap, and cursor etc. It is known that MiniGUI runtime configuration options is loaded from `MiniGUI.cfg`, but if compiling MiniGUI with in-core options, the options is included MiniGUI libraries.

In GNU development environment, after installing MiniGUI by default configuration, the file `etc/MiniGUI-classic.cfg` in MiniGUI source tree will be installed in `/usr/local/etc/` directory, and rename to `MiniGUI.cfg`. When MiniGUI application starts, the application first search `MiniGUI.cfg` in current directory, then search `.MiniGUI.cfg` in home directory, then search `MiniGUI.cfg` in `/usr/local/etc`, at last in `/etc/`. If user don't create the file `MiniGUI.cfg` in current directory and home directory, the application will use the file MiniGUI.cfg in `/usr/local/etc/` as default configuration file.

According to MiniGUI compiling configuration option, MiniGUI have four configuration files: `MiniGUI-classic.cfg`, `MiniGUI-fashion.cfg`, `MiniGUI-flat.cfg`, and `MiniGUI-min.cfg`. The details about MiniGUI compiling configuration options please refer to chapter 2.

When we compile MiniGUI with `--enable-incoreres` option, MiniGUI application doesn't need the file `MiniGUI.cfg`. The required options are given in the file `src/sysres/mgetc.c`.

Below, we first describe running configuration options with configuration file, and with incore resources.

3.1 Configuration File

The section describes configuration options in detail by `MiniGUI-classic.cfg`.

The format of configuration file is compact, and you can modify it easily. The following shows the format.

```
[section-name1]
```

```
key-name1=key-value1
key-name2=key-value2

[section-name2]
key-name3=key-value3
key-name4=key-value4
```

The parameters in the configuration file are grouped in sections, such as notation (#), section, key, and key value. The line that the first character is '#' is notation line. The values of the section are specified in the form of `section-name`. The values of the key and key value are specified in the form of `key=value`. Some important sections are listed as follows.

3.1.1 Section system

The section `system` not only defines the graphics engine (`gal_engine`) and the input engine (`ial_engine`) in runtime MiniGUI, which must be one of engines configured on MiniGUI compiling, but also defines the mouse device (`mdev`) and the mouse protocol type (`mtype`).

The definition of the keys in section `system` is as follows:

- `gal_engine`: The graphics engine used.
- `defaultmode`: The graphics engine display mode used, its format is widthxheight-bpp.
- `ial_engine`: The input engine used.
- `mdev`: The mouse device file.
- `mtype`: The mouse protocol type.

The contents of the section `system` in `MiniGUI.cfg` are as follow:

```
[system]
# GAL engine and default options
gal_engine=qvfb
defaultmode=800x600-16bpp

# IAL engine
ial_engine=qvfb

mdev=/dev/input/mice
mtype=IMPS2
```

Since MiniGUI Version 1.6.8, you can modify the graphics and input engine via environment variable. For example, if you define `fbcon` and `qvfb` graphics engine and `console` and `qvfb` input engine, and you choose the qvfb engine in `MiniGUI.cfg` or in-core resources. Then when configure MiniGUI, you can change the engine to fbcon and console in runtime by the following method, and needn’t modify **`MiniGUI.cfg`** or in-core resources configuration file.

```
$ export gal_engine=fbcon
$ export ial_engine=console
$ export mdev=/dev/input/mice
$ export mtype=ps2
$ export defaultmode=1024x768-16bpp
```

3.1.2 Section fbcon

The section **`fbcon`** is only available when you define the **`gal_engine`** in section **system** for fbcon. It define default display mode of the **`fbcon`** engine. When the section is undefined or key value is empty, the fbcon engine using the key value of system section.

The definition of the key in section **`fbcon`** is as follows:

- **`defaultmode`**: The display mode of graphics engine used, the format is **`widthxheight-bpp`**.

The content of the section in **`MiniGUI.cfg`** is as follows:

```
[fbcon]
defaultmode=1024x768-16bpp
```

3.1.3 Section qvfb

The section **`qvfb`** is only available when you define the **`gal_engine`** in section **`system`** for qvfb. It shows display and display mode of X window used when running qvfb.

The definition of the keys in section **`qvfb`** is as follows:

- **`defaultmode`**: The display mode of graphics engine used, its format is **`widthxheight-bpp`**.
- **`display`**: Display mode of X window used when running qvfb, default value is 0.

The content of the section in **`MiniGUI.cfg`** is as follows:

```
[qvfb]
defaultmode=640x480-16bpp
display=0
```

3.1.4 Section rawbitmapfonts, varbitmapfonts, qpf, truetypefonts, and type1fonts

These sections define information of loading `device fonts`, `number` of fonts, and name and file of fonts.

The format of device fonts used by MiniGUI is as follows:

```
<type>-<facename>-<style>-<width>-<height>-<charset1[,charset2,...]>
```

The definitions for each part of device **font** are as follow:

- **<type>**: The type of device font, for example, RBF, VBF, QPF, TrueType, and Adobe Type1 device font are rbf, vbf, qpf, ttf, and tlf.
- **<facename>**: The name of device font. Such as courier, Times etc.
- **<style>**: The style of device font, it is grouped into six alphabets. Such as bold, italic, underline or strikethrough etc. Generally the string is "rrncnn".
- **<width>**: The width of device font, for var-width fonts set to be maximum width; for vector fonts set to be 0.
- **<height>**: The height of device font, for vector fonts set to be 0.
- **<charset1, charset2>**: The charset of device font supported.

Each of these sections defines font_number, name**<NR>**, and fontfile**<NR>** keys.

- **font_number**: The number of device font loaded.
- **name<NR>**: The name of device font that number is **<NR>**.
- **fontfile<NR>**: The font file of device font that number is **<nr>**.

If you don't need to use a specific type of device font, you can skip the configuration option by set **font_number = 0**.

The content of these sections in **MiniGUI.cfg** are as follow:

```
[rawbitmapfonts]
font_number=4
name0=rbf-fixed-rrncnn-8-16-ISO8859-1
fontfile0=/usr/local/lib/minigui/res/font/8x16-iso8859-1.bin
name1=rbf-fixed-rrncnn-16-16-GB2312-0
fontfile1=/usr/local/lib/minigui/res/font/song-16-gb2312.bin
name2=rbf-fixed-rrncnn-6-12-ISO8859-1
fontfile2=/usr/local/lib/minigui/res/font/6x12-iso8859-1.bin
name3=rbf-fixed-rrncnn-12-12-GB2312-0
fontfile3=/usr/local/lib/minigui/res/font/song-12-gb2312.bin

[varbitmapfonts]
font_number=6
name0=vbf-Courier-rrncnn-8-13-ISO8859-1
fontfile0=/usr/local/lib/minigui/res/font/Courier-rr-8-13.vbf
name1=vbf-Helvetica-rrncnn-11-12-ISO8859-1
fontfile1=/usr/local/lib/minigui/res/font/Helvetica-rr-11-12.vbf
name2=vbf-Times-rrncnn-10-12-ISO8859-1
fontfile2=/usr/local/lib/minigui/res/font/Times-rr-10-12.vbf
name3=vbf-Courier-rrncnn-10-15-ISO8859-1
fontfile3=/usr/local/lib/minigui/res/font/Courier-rr-10-15.vbf
name4=vbf-Helvetica-rrncnn-15-16-ISO8859-1
```

```
fontfile4=/usr/local/lib/minigui/res/font/Helvetica-rr-15-16.vbf
name5=vbf-Times-rrncnn-13-15-ISO8859-1
fontfile5=/usr/local/lib/minigui/res/font/Times-rr-13-15.vbf

[qpf]
font_number=3
name0=qpf-unifont-rrncnn-16-16-ISO8859-1,ISO8859-15,GB2312-0,GBK,BIG5
fontfile0=/usr/local/lib/minigui/res/font/unifont_160_50.qpf
name1=qpf-times-rrncnn-5-10-ISO8859-1,ISO8859-15
fontfile1=/usr/local/lib/minigui/res/font/smoothtimes_100_50.qpf
name2=qpf-helvetica-rrncnn-5-10-ISO8859-1,ISO8859-15
fontfile2=/usr/local/lib/minigui/res/font/helvetica_100_50.qpf
name3=qpf-micro-rrncnn-4-4-ISO8859-1,ISO8859-15
fontfile3=/usr/local/lib/minigui/res/font/micro_40_50.qpf

[truetypefonts]
font_number=3
name0=ttf-arial-rrncnn-0-0-ISO8859-1
fontfile0=/usr/local/lib/minigui/res/font/arial.ttf
name1=ttf-times-rrncnn-0-0-ISO8859-1
fontfile1=/usr/local/lib/minigui/res/font/times.ttf
name2=ttf-pinball-rrncnn-0-0-ISO8859-1
fontfile2=/usr/local/lib/minigui/res/font/pinball.ttf

[type1fonts]
font_number=0
name0=type1-Charter-rrncnn-0-0-ISO8859-1
fontfile0=/usr/local/lib/minigui/res/font/bchr.pfb
name1=type1-Charter-rincnn-0-0-ISO8859-1
fontfile1=/usr/local/lib/minigui/res/font/bchri.pfb
name2=type1-Charter-brncnn-0-0-ISO8859-1
fontfile2=/usr/local/lib/minigui/res/font/bchb.pfb
name3=type1-Charter-bincnn-0-0-ISO8859-1
fontfile3=/usr/local/lib/minigui/res/font/bchbi.pfb
name4=type1-Courier-rrncnn-0-0-ISO8859-1
fontfile4=/usr/local/lib/minigui/res/font/dcr10.pfb
name5=type1-Courier-rincnn-0-0-ISO8859-1
fontfile5=/usr/local/lib/minigui/res/font/dcti10.pfb
name6=type1-Courier-brncnn-0-0-ISO8859-1
fontfile6=/usr/local/lib/minigui/res/font/dcbx10.pfb
name7=type1-Courier-bincnn-0-0-ISO8859-1
fontfile7=/usr/local/lib/minigui/res/font/dcbxti10.pfb
name8=type1-eufm10-rrncnn-0-0-ISO8859-1
fontfile8=/usr/local/lib/minigui/res/font/eufm10.pfb
```

3.1.5 Section systemfont

The section **`systemfont`** defines MiniGUI system font and font number, and defines system default font, which would be used to render text on captions, menus, and controls, as well as the default font of a window.

System font is the logic font[10] that is created by the function **`CreateLogFontFromName`** based on device fonts, which is defined by MiniGUI sections such as **`rawbitmapfonts`**, **`varbitmapfonts`**, **`qpf`**, **`truetypefonts`**, and **`t1fonts`**.

The content of the section in **`MiniGUI.cfg`** is as follows:

```
<type>-<facename>-<style>-<width>-<height>-<charset1>
```

The definition of each part of a logic font name is as follows:

- `<type>` is the desired device font type, if you do not want to specify it, use `*`.
- `<facename>` is to define the font face name, such as courier and times etc.
- `<style>` is the string of six alphabets to define style of a logic font, such as italic, bold, underline or strikethrough etc.
- `<width>` is to define the width of the logic font. Usually do not need to specify, use `*` instead.
- `<height>` is to define the height of the logic font.
- `<charset>` is to define charset of the logic font being created.

Many MiniGUI window matrices are defined based on the size of the default system font. Please refer to the explanation for section `mainwinmetric`.

Furthermore, MiniGUI V2.0.3/1.6.9 provides auto-scaling the font glyph. If you want to use this function, you only need use 'S' in forth character when you define logical font styles. Note that you don't need to use this style when you use vector font, such as TrueType, because vector font can produce corresponding font glyph according to desired logical font size.

The definition of the keys in section `systemfont` is as follows:

- `font_number`: The number of system fonts created
- `font<NR>`: The number <NR> logical font name
- `default`: System default font(single character set). Its value is the number of logical font.
- `wchar_def`: Default font used by multiple character set. Its value is the number of above logical font.
- `fixed`: The font used by fixed width character set. Its value is the number of above logical font.
- `caption`: The caption font. Its value is the number of above logical font.
- `menu`: The menu font. Its value is the number of above logical font.

You can change the number of system font created. But you must create a single character set (for example: ISO8859-1) at least. MiniGUI defines the system default charsets according to `default`, `wchar_def` system fonts, and this would affect the return value of `GetSysCharset`, `GetSysCharWidth`, `GetSysCCharWidth` and `GetSysHeight` functions. Commonly, `default` and `wchar_def` `must fixed width dot-matrix font, i.e RBF. And the width of multiply character set must be twice with the width of single character set.`

The content of the section in `MiniGUI.cfg` is as follows:

```
# The first system font must be a logical font using RBF device font.
[systemfont]
font_number=6
font0=rbf-fixed-rrncnn-8-16-ISO8859-1
font1=*-fixed-rrncnn-*-16-GB2312
font2=*-Courier-rrncnn-*-16-GB2312
font3=*-SansSerif-rrncnn-*-16-GB2312
font4=*-Times-rrncnn-*-16-GB2312
font5=*-Helvetica-rrncnn-*-16-GB2312

default=0
wchar_def=1
fixed=1
caption=2
menu=3
control=3
```

3.1.6 Section mouse

The section `mouse` defines the time of mouse `double clicked`. It is used to handle with system inner events. Generally, it is unnecessary changed.

The definition of the keys in the section is as follows:

- `dblclicktime`: The mouse double clicked time in ms

The content of the section in `MiniGUI.cfg` is as follows:

```
[mouse]
dblclicktime=300
```

3.1.7 Section event

The section `event` defines event timeout and auto-repeat time used by system internal event process. Generally, it is unnecessary changed.

The definition of the keys in the section is as follows:

- `timeoutusec`: Event timeout time in ms
- `repeatusec`: Event repeat time in ms

The content of the section in `MiniGUI.cfg` is as follows:

```
timeoutusec=300000
repeatusec=50000
```

3.1.8 Section cursorinfo

This section defines information for `mouse cursor` loaded by MiniGUI.

If you use `--disable-cursor` to compile configuration option, the MiniGUI ignore cursorinfo section.

The definition of the keys in the section is as follows:

- **`cursorpath`**: The path for cursor file
- **`cursornumber`**: The number of cursor loaded. It can save the store space by reducing cursor number and deleting cursor file.
- **`cursor<NR>`**: the cursor that number is **`<NR>`**

The content of the section in **`MiniGUI.cfg`** is as follows:

```
[cursorinfo]
# Edit following line to specify cursor files path
cursorpath=/usr/local/lib/minigui/res/cursor/
cursornumber=23
cursor0=d_arrow.cur
cursor1=d_beam.cur
cursor2=d_pencil.cur
cursor3=d_cross.cur
cursor4=d_move.cur
cursor5=d_sizenwse.cur
cursor6=d_sizenesw.cur
cursor7=d_sizewe.cur
cursor8=d_sizens.cur
cursor9=d_uparrow.cur
cursor10=d_none.cur
cursor11=d_help.cur
cursor12=d_busy.cur
cursor13=d_wait.cur
cursor14=g_rarrow.cur
cursor15=g_col.cur
cursor16=g_row.cur
cursor17=g_drag.cur
cursor18=g_nodrop.cur
cursor19=h_point.cur
cursor20=h_select.cur
cursor21=ho_split.cur
cursor22=ve_split.cur
```

3.1.9 Section iconinfo

The section **`iconinfo`** defines MiniGUI icon information loaded.

The definition of the keys in the section is as follows:

- **`iconpath`**: The path of the icon.
- **`iconnumber`**: The number of icon loaded, maximum is 5. You can decrease the number.
- **`icon<NR>`**: The number <NR>'s icon.
- **`fold`**, **`unfold`**: Default icon used by TreeView control.
- **`dir`**, **`file`**: Default icon used by OpenFileDialogBox control.

The content of the section in **`MiniGUI.cfg`** is as follows:

```
[iconinfo]
# Edit following line to specify icon files path
iconpath=/usr/local/lib/minigui/res/icon/
# Note that max number defined in source code is 5.
iconnumber=5
icon0=form.ico
icon1=failed.ico
icon2=mg_help.ico
```

```
icon3=warning.ico
icon4=excalmatory.ico

# default icons for TREEVIEW control
fold=fold.ico
unfold=unfold.ico

# default icons for new OpenFileDialogBox
dir=folder.ico
file=textfile.ico
```

3.1.10 Section bitmapinfo

This section defines information of `bitmap` loaded.

The definition of the keys in the section is as follows:

- `bitmappath`: The path for bitmap file.
- `bitmapnumber`: The number of bitmap loaded, and the maximum value is 7. It can save the store space by reducing bitmap number and deleting bitmap file.
- `bitmap<NR>`: The number of bitmap that number is `<NR>`.
- `caption`: The image of window caption bar, only it is enable for fashion style.
- `checkmark`: The bitmap for checkbox in Listbox control.
- `downarrow`: The bitmap for down arrow in ComboBox control.
- `updownarrow`: The bitmap for up or down arrow in ComboBox control.
- `leftrightarrow`: The bitmap for left or right arrow in ComboBox control.
- `spinbox_vert`: The bitmap for vertical arrow in SpinBox control.
- `spinbox_horz`: The bitmap for horizontal arrow in SpinBox control.
- `IMEctrlbtn`: The bitmap for IME window. You can delete the bitmap, if you use `--disable-imegb2312` compiling configuration to compile MiniGUI.
- `logo`: The bitmap for about dialog. You can delete the bitmap, if you use `--disable-aboutdlg` compiling configuration to compile MiniGUI.

The content of the section in `MiniGUI.cfg` is as follows:

```
[bitmapinfo]
# Edit following line to specify bitmap files path
bitmappath=/usr/local/lib/minigui/res/bmp/
# Note that max number defined in source code is 7
bitmapnumber=2
bitmap0=capbtns.bmp
bitmap1=arrows.bmp
# use large bitmap if your default font is 16 pixel height.
# bitmap1=arrows16.bmp
bitmap2=none
bitmap3=none
bitmap4=none
bitmap5=none

# bitmap used by captionbar of main window (only Fashion style)
caption=

# bitmap used by BUTTON control
```

```
button=button.bmp

# bitmap used by LISTBOX control
checkmark=checkmark.bmp

# bitmap used by COMBOBOX control
downarrow=downarrow.bmp
updownarrow=updownarrow.bmp
leftrightarrow=leftrightarrow.bmp


# bitmap used by SPINBOX control
spinbox_vert=spinbox-vert.bmp
spinbox_horz=spinbox-horz.bmp

# bitmap used by listview control
lvfold=lvfold.bmp
lvunfold=lvunfold.bmp

# bitmap used by IME window
IMEctrlbtn=shurufa.bmp

# bitmap used by About dialog box
logo=MiniGUI256.bmp
# logo=MiniGUI16.bmp
```

3.1.11 Section bgpicture

The section **bgpicture** defines MiniGUI desktop background picture and display position. The definition of the keys in the section is as follows:

- **file**: The whole path of background picture. If no background picture, it should be none.
- **Position**: The display position of background picture, its value can be one of center, upleft, downleft, upright, downright, upcenter, downcenter, vcenterleft, vcenterright, and none.

The content of the section in **MiniGUI.cfg** is as follows:

```
The definition of the section bgpicture in MiniGUI.cfg is as follows
# background picture, use your favirate photo
file=none
position=center
# position=upleft
# position=downleft
# position=upright
# position=downright
# position=upcenter
# position=downcenter
# position=vcenterleft
# position=vcenterright
# position=none
```

3.1.12 Section mainwinmetrics

This section defines default size of **main window**. Generally, it is unnecessary changed.

We can set window size with height of system font. For example, we can set the height of caption with **captiony=+4** which means to set the height of caption to the height of system font plus 4.

The definition of the keys in the section is as follows:

- **`minwidth`**: The minimal width of a main window.
- **`minheight`**: The minimal height of a main window.
- **`border`**: The width of the border of a main window.
- **`thickframe`**: The width of the thick frame of a main window.
- **`thinframe`**: The width of the thin frame of a main window.
- **`captiony`**: The height of the caption of a main window.
- **`iconx`**: The X coordinate of the icon.
- **`icony`**: The Y coordinate of the icon.
- **`menubary`**: The height of menu bar.
- **`menubaroffx`**: The horizontal offset of menu bar.
- **`menubaroffy`**: The vertical offset of menu bar.
- **`menuitemy`**: The height of menu item.
- **`intermenuitemx`**: The horizontal distance between two menu item.
- **`intermenuitemy`**: The vertical distance between two menu item.
- **`menuitemoffx`**: The horizontal offset of menu item.
- **`menutopmargin`**: The top margin of a menu item.
- **`menubottommargin`**: The bottom margin of a menu item.
- **`menuleftmargin`**: The left margin of a menu item.
- **`menurightmargin`**: The right margin of a menu item.
- **`menuitemminx`**: The minimal width of a menu item.
- **`menuseparatory`**: The height of a separator menu item.
- **`menuseparatorx`**: The width of a separator menu item.
- **`sb_height`**: The height of the button on the scroll bar.
- **`sb_width`**: The width of the button on the scroll bar.
- **`sb_interx`**: The width of a horizontal scroll bar.
- **`cxvscroll`**: The width of a vertical scroll bar.
- **`cyvscroll`**: The height of a vertical scroll bar.
- **`cxhscroll`**: The width of a horizontal scroll bar.
- **`cyhscroll`**: The height of a horizontal scroll bar.
- **`minbarlen`**: The minimal track-bar length of the scroll bar.
- **`defbarlen`**: The default track-bar length of the scroll bar.

The content of the section in **`MiniGUI.cfg`** is as follows:

```
[mainwinmetrics]
minwidth=50
minheight=50
border=2
thickframe=2
thinframe=1
captiony=+4
iconx=16
icony=16
menubary=+0
menubaroffx=8
menubaroffy=5
menuitemy=+0
intermenuitemx=12
intermenuitemy=2
menuitemoffx=18
menutopmargin=4
menubottommargin=4
menuleftmargin=4
menurightmargin=4
menuitemminx=64
menuseparatory=4
menuseparatorx=4
sb_height=14
sb_width=16
sb_interx=2
cxvscroll=12
cyvscroll=12
cxhscroll=12
cyhscroll=12
minbarlen=9
defbarlen=18
```

3.1.13 Section windowelementcolors

The section **windowelementcolors** defines default window element color used. Generally, it is unnecessary changed.

The definition of the keys in section is as follows:

- **bkc_caption_normal**: The background color of the normal caption.
- **fgc_caption_normal**: The foreground color of the normal caption.
- **bkc_caption_actived**: The background color of the active caption.
- **fgc_caption_actived**: The foreground color of the active caption.
- **bkc_caption_disabled**: The background color of the disabled caption.
- **fgc_caption_disabled**: The foreground color of the disabled caption.
- **wec_frame_normal**: The color of the normal frame.
- **wec_frame_actived**: The color of the active frame.
- **wec_frame_disabled**: The color of the disabled frame.
- **bkc_menubar_normal**: The background color of the normal menubar.
- **fgc_menubar_normal**: The foreground color of the normal menubar.
- **bkc_menubar_hilite**: The background color of the highlight menubar.
- **fgc_menubar_hilite**: The foreground color of the highlight menubar.

- `fgc_menubar_disabled`: The foreground color of the disabled menubar.
- `bkc_menuitem_normal`: The background color of the normal menubar.
- `fgc_menuitem_normal`: The foreground color of the normal menubar.
- `bkc_menuitem_hilite`: The background color of the highlight menubar.
- `fgc_menuitem_hilite`: The foreground color of the highlight menubar.
- `fgc_menuitem_disabled`: The foreground color of the disabled menubar.
- `bkc_pppmenutitle`: The background color of the popup menu title.
- `fgc_pppmenutitle`: The foreground color of the popup menu title.
- `fgc_menuitem_frame`: The foreground color of the menuitem frame.
- `wec_3dbox_normal`: The color of the normal 3dbox.
- `wec_3dbox_reverse`: The color of the reversed 3dbox.
- `wec_3dbox_light`: The color of the highlight 3dbox.
- `wec_3dbox_dark`: The color of the dark 3dbox.
- `wec_flat_border`: The color of the border with Flat style.
- `bkc_control_def`: The background color of default control.
- `fgc_control_normal`: The foreground color of the normal control.
- `fgc_control_disabled`: The foreground color of the disabled control .
- `bkc_hilight_normal`: The background color of the normal highlight control .
- `bkc_hilight_lostfocus`: The foreground color of the highlight control after losing focus.
- `fgc_hilight_normal`: The foreground color of the normal highlight control.
- `fgc_hilight_disabled`: The foreground color of the disabled highlight control.
- `bkc_desktop`: The background color of the desktop.
- `bkc_dialog`: The background color of the default dialog.
- `bkc_tip`: The background color of the tip.

The content of the section in `MiniGUI.cfg` is as follows:

```
bkc_caption_normal=0x00808080
fgc_caption_normal=0x00C8D0D4
bkc_caption_actived=0x006A240A
fgc_caption_actived=0x00FFFFFF
bkc_caption_disabled=0x00808080
fgc_caption_disabled=0x00C8D0D4

wec_frame_normal=0x00FFFFFF
wec_frame_actived=0x00FFFFFF
wec_frame_disabled=0x003704EA

bkc_menubar_normal=0x00CED3D6
fgc_menubar_normal=0x00000000
bkc_menubar_hilite=0x003704EA
fgc_menubar_hilite=0x00FFFFFF
fgc_menubar_disabled=0x00848284
bkc_menuitem_normal=0x00CED3D6
fgc_menuitem_normal=0x00000000
```

```
bkc_menuitem_hilite=0x006B2408
fgc_menuitem_hilite=0x00FFFFFF
fgc_menuitem_disabled=0x00848284

bkc_pppmenutitle=0x00C0C0C0
fgc_pppmenutitle=0x006B2408
fgc_menuitem_frame=0x00C66931

wec_3dbox_normal=0x00CED3D6
wec_3dbox_reverse=0x00000000
wec_3dbox_light=0x00FFFFFF
wec_3dbox_dark=0x00808080
wec_flat_border=0x00808080

bkc_control_def=0x00CED3D6
fgc_control_normal=0x00000000
fgc_control_disabled=0x00848284

bkc_hilight_normal=0x006B2408
bkc_hilight_lostfocus=0x00BDA69C
fgc_hilight_normal=0x00FFFFFF
fgc_hilight_disabled=0x00C0C0C0
bkc_desktop=0x00C08000
bkc_dialog=0x00CED3D6
bkc_tip=0x00E7FFFF
```

3.1.14 Section imeinfo

This section defines the number and module about **`GB2312 IME`**.

The definition of the keys in the section is as follows:

- **`imetabpath`**: The path of IME module.
- **`imenumber`**: The number of IME module.
- **`ime<NR>`**: The IME module that number is <NR>. Pinyin is pinyin module. wubi is wubi module. shuangpin is shuangpin module. ziranma is ziranma module etc.

If we enable GB2312 IME (--enable-imegb2312) in configuration, the MiniGUI will load the IME module with imenumber value, if imenumber is 0, the MiniGUI only include ISA IME module.

The content of the section in **`MiniGUI.cfg`** is as follows:

```
imetabpath=/usr/local/lib/minigui/res/imetab/
imenumber=1
ime0=pinyin
```

3.1.15 Default Configuration File

Below is the default runtime configuration file for MiniGUI library:

```
# MiniGUI Ver 2.0.x
# This configuration file is for classic window style.
#
# Copyright (C) 2002~2007 Feynman Software
# Copyright (C) 1998~2002 Wei Yongming.
#
```

```
# Web:   http://www.minigui.com
# Web:   http://www.minigui.org
#
# This configuration file must be installed in /etc,
# /usr/local/etc or your home directory. When you install it in your
# home directory, it should be named ".MiniGUI.cfg".
#
# The priority of above configruation files is ~/.MiniGUI.cfg,
# /usr/local/etc/MiniGUI.cfg, and then /etc/MiniGUI.cfg.
#
# If you change the install path of MiniGUI resource, you should
# modify this file to meet your configuration.
#
# NOTE:
# The format of this configuration file has changed since the last release.
# Please DONT forget to provide the latest MiniGUI.cfg file for your MiniGUI.
#

[system]
# GAL engine and default options
gal_engine=qvfb
defaultmode=800x600-16bpp

# IAL engine
ial_engine=qvfb
mdev=/dev/input/mice
mtype=IMPS2

[fbcon]
defaultmode=1024x768-16bpp

[qvfb]
defaultmode=640x480-16bpp
display=0

# The first system font must be a logical font using RBF device font.
[systemfont]
font_number=6
font0=rbf-fixed-rrncnn-8-16-ISO8859-1
font1=*-fixed-rrncnn-*-16-GB2312
font2=*-Courier-rrncnn-*-16-GB2312
font3=*-SansSerif-rrncnn-*-16-GB2312
font4=*-Times-rrncnn-*-16-GB2312
font5=*-Helvetica-rrncnn-*-16-GB2312

default=0
wchar_def=1
fixed=1
caption=2
menu=3
control=3

[rawbitmapfonts]
font_number=4
name0=rbf-fixed-rrncnn-8-16-ISO8859-1
fontfile0=/usr/local/lib/minigui/res/font/8x16-iso8859-1.bin
name1=rbf-fixed-rrncnn-16-16-GB2312-0
fontfile1=/usr/local/lib/minigui/res/font/song-16-gb2312.bin
name2=rbf-fixed-rrncnn-6-12-ISO8859-1
fontfile2=/usr/local/lib/minigui/res/font/6x12-iso8859-1.bin
name3=rbf-fixed-rrncnn-12-12-GB2312-0
fontfile3=/usr/local/lib/minigui/res/font/song-12-gb2312.bin

[varbitmapfonts]
font_number=6
name0=vbf-Courier-rrncnn-8-13-ISO8859-1
fontfile0=/usr/local/lib/minigui/res/font/Courier-rr-8-13.vbf
name1=vbf-Helvetica-rrncnn-11-12-ISO8859-1
fontfile1=/usr/local/lib/minigui/res/font/Helvetica-rr-11-12.vbf
name2=vbf-Times-rrncnn-10-12-ISO8859-1
fontfile2=/usr/local/lib/minigui/res/font/Times-rr-10-12.vbf
name3=vbf-Courier-rrncnn-10-15-ISO8859-1
fontfile3=/usr/local/lib/minigui/res/font/Courier-rr-10-15.vbf
name4=vbf-Helvetica-rrncnn-15-16-ISO8859-1
fontfile4=/usr/local/lib/minigui/res/font/Helvetica-rr-15-16.vbf
```

```
name5=vbf-Times-rrncnn-13-15-ISO8859-1
fontfile5=/usr/local/lib/minigui/res/font/Times-rr-13-15.vbf

[qpf]
font_number=3
name0=qpf-unifont-rrncnn-16-16-ISO8859-1,ISO8859-15,GB2312-0,GBK,BIG5
fontfile0=/usr/local/lib/minigui/res/font/unifont_160_50.qpf
name1=qpf-times-rrncnn-5-10-ISO8859-1,ISO8859-15
fontfile1=/usr/local/lib/minigui/res/font/smoothtimes_100_50.qpf
name2=qpf-helvetica-rrncnn-5-10-ISO8859-1,ISO8859-15
fontfile2=/usr/local/lib/minigui/res/font/helvetica_100_50.qpf
name3=qpf-micro-rrncnn-4-4-ISO8859-1,ISO8859-15
fontfile3=/usr/local/lib/minigui/res/font/micro_40_50.qpf

[truetypefonts]
font_number=3
name0=ttf-arial-rrncnn-0-0-ISO8859-1
fontfile0=/usr/local/lib/minigui/res/font/arial.ttf
name1=ttf-times-rrncnn-0-0-ISO8859-1
fontfile1=/usr/local/lib/minigui/res/font/times.ttf
name2=ttf-pinball-rrncnn-0-0-ISO8859-1
fontfile2=/usr/local/lib/minigui/res/font/pinball.ttf

[type1fonts]
font_number=0
name0=type1-Charter-rrncnn-0-0-ISO8859-1
fontfile0=/usr/local/lib/minigui/res/font/bchr.pfb
name1=type1-Charter-rincnn-0-0-ISO8859-1
fontfile1=/usr/local/lib/minigui/res/font/bchri.pfb
name2=type1-Charter-brncnn-0-0-ISO8859-1
fontfile2=/usr/local/lib/minigui/res/font/bchb.pfb
name3=type1-Charter-bincnn-0-0-ISO8859-1
fontfile3=/usr/local/lib/minigui/res/font/bchbi.pfb
name4=type1-Courier-rrncnn-0-0-ISO8859-1
fontfile4=/usr/local/lib/minigui/res/font/dcr10.pfb
name5=type1-Courier-rincnn-0-0-ISO8859-1
fontfile5=/usr/local/lib/minigui/res/font/dcti10.pfb
name6=type1-Courier-brncnn-0-0-ISO8859-1
fontfile6=/usr/local/lib/minigui/res/font/dcbx10.pfb
name7=type1-Courier-bincnn-0-0-ISO8859-1
fontfile7=/usr/local/lib/minigui/res/font/dcbxti10.pfb
name8=type1-eufm10-rrncnn-0-0-ISO8859-1
fontfile8=/usr/local/lib/minigui/res/font/eufm10.pfb

[mouse]
dblclicktime=300

[event]
timeoutusec=300000
repeatusec=50000

[cursorinfo]
# Edit following line to specify cursor files path
cursorpath=/usr/local/lib/minigui/res/cursor/
cursornumber=23
cursor0=d_arrow.cur
cursor1=d_beam.cur
cursor2=d_pencil.cur
cursor3=d_cross.cur
cursor4=d_move.cur
cursor5=d_sizenwse.cur
cursor6=d_sizenesw.cur
cursor7=d_sizewe.cur
cursor8=d_sizens.cur
cursor9=d_uparrow.cur
cursor10=d_none.cur
cursor11=d_help.cur
cursor12=d_busy.cur
cursor13=d_wait.cur
cursor14=g_rarrow.cur
cursor15=g_col.cur
cursor16=g_row.cur
cursor17=g_drag.cur
cursor18=g_nodrop.cur
cursor19=h_point.cur
```

```
cursor20=h_select.cur
cursor21=ho_split.cur
cursor22=ve_split.cur

[iconinfo]
# Edit following line to specify icon files path
iconpath=/usr/local/lib/minigui/res/icon/
# Note that max number defined in source code is 5.
iconnumber=5
icon0=form.ico
icon1=failed.ico
icon2=mg_help.ico
icon3=warning.ico
icon4=excalmatory.ico

# default icons for TREEVIEW control
fold=fold.ico
unfold=unfold.ico

# default icons for new OpenFileDialogBox
dir=folder.ico
file=textfile.ico

[bitmapinfo]
# Edit following line to specify bitmap files path
bitmappath=/usr/local/lib/minigui/res/bmp/
# Note that max number defined in source code is 7
bitmapnumber=2
bitmap0=capbtns.bmp
bitmap1=arrows.bmp
# use large bitmap if your default font is 16 pixel height.
# bitmap1=arrows16.bmp
bitmap2=none
bitmap3=none
bitmap4=none
bitmap5=none

# bitmap used by captionbar of main window (only Fashion style)
caption=

# bitmap used by BUTTON control
button=button.bmp

# bitmap used by LISTBOX control
checkmark=checkmark.bmp

# bitmap used by COMBOBOX control
downarrow=downarrow.bmp
updownarrow=updownarrow.bmp
leftrightarrow=leftrightarrow.bmp


# bitmap used by SPINBOX control
spinbox_vert=spinbox-vert.bmp
spinbox_horz=spinbox-horz.bmp

# bitmap used by listview control
lvfold=lvfold.bmp
lvunfold=lvunfold.bmp

# bitmap used by IME window
IMEctrlbtn=shurufa.bmp

# bitmap used by About dialog box
logo=MiniGUI256.bmp
# logo=MiniGUI16.bmp

[bgpicture]
# background picture, use your favirate photo
file=none
position=center
# position=upleft
# position=downleft
# position=upright
# position=downright
# position=upcenter
# position=downcenter
# position=vcenterleft
# position=vcenterright
# position=none

[mainwinmetrics]
minwidth=50
minheight=50
border=2
thickframe=2
thinframe=1
captiony=+4
iconx=16
icony=16
menubary=+0
menubaroffx=8
menubaroffy=5
menuitemy=+0
intermenuitemx=12
intermenuitemy=2
menuitemoffx=18
menutopmargin=4
menubottommargin=4
menuleftmargin=4
menurightmargin=4
menuitemminx=64
menuseparatory=4
menuseparatorx=4
sb_height=14
sb_width=16
sb_interx=2
cxvscroll=12
cyvscroll=12
cxhscroll=12
cyhscroll=12
minbarlen=9
defbarlen=18

[windowelementcolors]
bkc_caption_normal=0x00808080
fgc_caption_normal=0x00C8D0D4
bkc_caption_actived=0x006A240A
fgc_caption_actived=0x00FFFFFF
bkc_caption_disabled=0x00808080
fgc_caption_disabled=0x00C8D0D4

wec_frame_normal=0x00FFFFFF
wec_frame_actived=0x00FFFFFF
wec_frame_disabled=0x003704EA

bkc_menubar_normal=0x00CED3D6
fgc_menubar_normal=0x00000000
bkc_menubar_hilite=0x003704EA
fgc_menubar_hilite=0x00FFFFFF
fgc_menubar_disabled=0x00848284
bkc_menuitem_normal=0x00CED3D6
fgc_menuitem_normal=0x00000000
bkc_menuitem_hilite=0x006B2408
fgc_menuitem_hilite=0x00FFFFFF
fgc_menuitem_disabled=0x00848284

bkc_pppmenutitle=0x00C0C0C0
fgc_pppmenutitle=0x006B2408
fgc_menuitem_frame=0x00C66931

wec_3dbox_normal=0x00CED3D6
wec_3dbox_reverse=0x00000000
wec_3dbox_light=0x00FFFFFF
wec_3dbox_dark=0x00808080
wec_flat_border=0x00808080

bkc_control_def=0x00CED3D6
fgc_control_normal=0x00000000
fgc_control_disabled=0x00848284
```

```
bkc_hilight_normal=0x006B2408
bkc_hilight_lostfocus=0x00BDA69C
fgc_hilight_normal=0x00FFFFFF
fgc_hilight_disabled=0x00C0C0C0
bkc_desktop=0x00C08000
bkc_dialog=0x00CED3D6
bkc_tip=0x00E7FFFF

[imeinfo]
imetabpath=/usr/local/lib/minigui/res/imetab/
imenumber=1
ime0=pinyin
```

3.2 Incore Configuration Options

When use incore resources, MiniGUI don't need the file **`MiniGUI.cfg`**. The appropriate configuration options are defined in the file **`src/sysres/mgetc.c`**.

Similar with the structure in **`MiniGUI.cfg`**, MiniGUI defines an structure ETCSECTION, array **`mgetc_sections`** and variable MGETC in **`mgetc.c`**. The array mgetc_sections is appropriate with section in configuration file. MGETC that is ETC_S type is appropriate with configuration file.

3.2.1 Structure ETCSETCTION

The structure **`ETCSECTION`** is defined in the file named '**`minigui.h`**'. The following is in detail.

```
/** Etc The current config section information */
typedef struct _ETCSECTION
{
    /** Allocated number of keys */
    int key_nr_alloc;
    /** Key number in the section */
    int key_nr;
    /** Name of the section */
    char *name;
    /** Array of keys */
    char** keys;
    /** Array of values */
    char** values;
} ETCSECTION;
```

The **`key_nr_alloc`** is the interface of other configuration options. Its value must be 0 in incore. The **`key_nr`** defines the number of the key in section. The name defines the name of section. The keys and values is the array of key and value. The number of key array and value array is corresponded with the number of the **`key_nr`**.

Below is the definition of mgetc_sections in the **`mgetc.c`** file.

```
static ETCSECTION mgetc_sections [] =
{
  {0, 4, "system",       SYSTEM_KEYS,     SYSTEM_VALUES},
  {0, 2, "qvfb",         QVFB_KEYS,       QVFB_VALUES},
  {0, 2, "wvfb",         FBCON_KEYS,      FBCON_VALUES},
  {0, 2, "shadow",       FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "fbcon",        FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "dummy",        FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "em85xxyuv",    FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "em85xxosd",    FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "svpxxosd",     FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "utpmc",        FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "bf533",        FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "mb93493",      FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "commlcd",      FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "dfb",          FBCON_KEYS,      FBCON_VALUES},
  {0, TABLESIZE(SYSTEMFONT_KEYS), "systemfont",  SYSTEMFONT_KEYS, SYSTEMFONT_VALUES},
  {0, 1, "cursorinfo",   CURSORINFO_KEYS, CURSORINFO_VALUES},
  {0, 1, "iconinfo",     ICONINFO_KEYS,   ICONINFO_VALUES},
  {0, 1, "bitmapinfo",   BITMAPINFO_KEYS, BITMAPINFO_VALUES},
/* optional sections */
  {0, 2, "bgpicture", BGPICTURE_KEYS, BGPICTURE_VALUES},
  /*
  {1, "mouse", MOUSE_KEYS, MOUSE_VALUES},
  */
  {0, 2, "event", EVENT_KEYS, EVENT_VALUES},
#ifdef _IME_GB2312
  {0, 3, "imeinfo", IMEINFO_KEYS, IMEINFO_VALUES},
#endif
#if defined (_TTF_SUPPORT) || defined (_FT2_SUPPORT)
  {0, TABLESIZE(TTFINFO_KEYS), "truetypefonts", TTFINFO_KEYS, TTFINFO_VALUES},
#endif
};
```

The section in **`mgetc_sections`** must be defined (fbcon or qvfb is optional.). Other notation sections are optional. The meaning of sections is same as the sections in MiniGUI.cfg. Commonly, you can only change the GAL engine, the IAL engine, display mode and the sections of system and fbcon: SYSTEM_VALUES and FBCON_VALUES defined in the **`mgetc-xxx.c`** file, such as **`mgetc-pc.c`**.

The **`systemfont`** section defines incore font used by system. Currently, MiniGUI 2.0.x supports ISO8859-1, GB2312, RBF, BIG5, SHIFT_JIS, and QPF. MiniGUI doesn't support the TTF and Type1 font in incore resources.

3.2.2 ETC_S Structure

`ETC_S` structure was defined in the file minigui.h, the content of ETC_S listed as the follow:

```
/** ETC_S The current config file information*/
typedef struct _ETC_S
{
    /** Allocated number of sections */
    int sect_nr_alloc;
    /** Number of sections */
    int section_nr;
    /** Pointer to section arrays */
    PETCSECTION sections;
} ETC_S;
```

Therefore, **sect_nr_alloc** is the interface of the other configuration options, it's value must be 0 in incore, **sect_nr** specify the number of section, sections is ETCSECTION type structure array, the number of item is not less than the value, the first item specified this value.

The **mgetc_sections** array was defined as the follow in the **mgetc.c** file.

```
ETC_S MGETC = {0, TABLESIZE (mgetc_sections), mgetc_sections};
```

The number of section is TABLESIZE (mgetc_sections) in the MGETC structure; the section array is mgetc_sections array above.

3.2.3 Listing of mgetc.c

```
/*
 ** $Id: mgetc.c 7249 2007-06-06 06:36:42Z weiym $
 **
 ** mgetc.c: definitions for incore resource.
 **
 ** Copyright (C) 2003 ~ 2007 Feynman Software.
 **
 ** Create date: 2003/09/22
 */
#include <stdio.h>

#include "common.h"
#include "minigui.h"

#ifdef _INCORE_RES

#define _ETC_CONFIG_EVENT

#ifndef DYNAMIC_LOAD

#ifdef _CUSTOM_IAL
/*
 * Please modify this file to meet your customer's board
 * system configuration options.
 */
#include "mgetc-custom.c"
#endif

#ifdef __VXWORKS__
/*
 * System configure files for boards running VxWorks.
 * Please move the file which you use to be the first file in this group.
 */
#include "mgetc-vxi386.c"
#include "mgetc-vxwifi.c"
#include "mgetc-vxsim.c"
#include "mgetc-vxppc.c"
#endif

#ifdef __UCOSII__
/*
 * System configure files for boards running uC/OS-II.
 * Please move the file which you use to be the first file in this group.
 */
#include "mgetc-ucosii-arm3000.c"
#include "mgetc-ucosii-skyeye.c"
#endif

#ifdef __ECOS__
/*
 * System configure files for boards running eCos.
 * Please move the file which you use to be the first file in this group.
 */
#include "mgetc-ecos-default.c"
#include "mgetc-ecos-ipaq-wifi.c"
#include "mgetc-ecos-palm2.c"
#endif

#ifdef __WINBOND_SWLINUX__
#include "mgetc-swlinux.c"
#endif

#ifdef __CYGWIN__
#include "mgetc-cygwin.c"
#endif

#ifdef __WIN32__
#include "mgetc-win32.c"
#endif

#ifdef __THREADX__
#include "mgetc-threadx.c"
#endif

#ifdef __NUCLEUS__
/*
 * System configure files for boards running Nucleus.
 * Please move the file which you use to be the first file in this group.
 */
#include "mgetc-nucleus.c"
#include "mgetc-nucleus-mnt.c"
#include "mgetc-nucleus-monaco.c"
#endif

#ifdef __PSOS__
#include "mgetc-psos-default.c"
#endif

#ifdef __OSE__
#include "mgetc-ose-mx21.c"
#endif

#ifdef __uClinux__
/*
 * System configure files for boards running uClinux
 * Please move the file which you use to be the first file in this group.
 */
#include "mgetc-bfin.c"
#include "mgetc-axlinux.c"
#include "mgetc-bf533.c"
#include "mgetc-em86.c"
#include "mgetc-em85.c"
#include "mgetc-hh44b0.c"
#include "mgetc-uptech.c"
#include "mgetc-mb93493.c"
#include "mgetc-utpmc.c"
#endif

#ifndef _SYS_CFG_INCLUDED
/* system configure files for boards running Linux */
#   ifdef _IPAQ_IAL
#include "mgetc-ipaq.c"
#   endif
#   ifdef _FIGUEROA_IAL
#include "mgetc-figueroa.c"
#   endif
#   ifdef _FFT7202_IAL
#include "mgetc-fft7202.c"
```

```
#   endif
#   ifdef _DM270_IAL
#include "mgetc-dm270.c"
#   endif
#   ifdef _EVMV10_IAL
#include "mgetc-xscale.c"
#   endif
#   ifdef _EMBEST2410_IAL
#include "mgetc-embest2410.c"
#   endif
#   ifdef _FXRM9200_IAL
#include "mgetc-rm9200.c"
#   endif
#   ifdef _HH2410R3_IAL
#include "mgetc-hh2410r3.c"
#   endif
#   ifdef _HH2410R3_IAL
#include "mgetc-hh2440.c"
#   endif

#include "mgetc-pc.c"

#endif /* !_SYS_CFG_INCLUDED */

static char *SYSTEM_KEYS[] = {"gal_engine", "ial_engine", "mdev", "mtype"};

static char *FBCON_KEYS[] = {"defaultmode"};

static char *QVFB_KEYS[] = {"defaultmode", "display"};
static char *QVFB_VALUES[] = {"640x480-16bpp", "0"};

static char *CURSORINFO_KEYS[] = {"cursornumber"};
static char *CURSORINFO_VALUES[] = {"23"};

static char *ICONINFO_KEYS[] = {"iconnumber"};
static char *ICONINFO_VALUES[] = {"5"};

static char *BITMAPINFO_KEYS[] = {"bitmapnumber"};
static char *BITMAPINFO_VALUES[] = {"6"};
static char *BGPICTURE_KEYS[] = {"position", "file"};

#ifdef BACKGROUND_IMAGE_FILE
static char *BGPICTURE_VALUES[] = {"center", BACKGROUND_IMAGE_FILE};
#else
static char *BGPICTURE_VALUES[] = {"none", ""};
#endif

#ifdef _ETC_CONFIG_EVENT
static char *EVENT_KEYS[] = {"timeoutusec", "repeatusec"};
static char *EVENT_VALUES[] = {"300000", "50000"};
#endif

#ifdef _IME_GB2312
static char* IMEINFO_KEYS[] = {"imetabpath", "imenumber", "ime0"};
static char* IMEINFO_VALUES[] = {"/usr/local/lib/minigui/res/imetab/", "1", "pinyin"};
#endif

#if defined(_TTF_SUPPORT) || defined(_FT2_SUPPORT)
static char* TTFINFO_KEYS[] = {"font_number", "name0", "fontfile0", "name1", "fontfile1"};
static char* TTFINFO_VALUES[] = {"2",
        "ttf-arial-rrncnn-0-0-ISO8859-1", "/usr/local/lib/minigui/res/font/arial.ttf",
        "ttf-times-rrncnn-0-0-ISO8859-1", "/usr/local/lib/minigui/res/font/times.ttf"
};
#endif

static ETCSECTION mgetc_sections [] =
{
  {0, 4, "system",       SYSTEM_KEYS,     SYSTEM_VALUES},
  {0, 2, "qvfb",         QVFB_KEYS,       QVFB_VALUES},
  {0, 2, "wvfb",         FBCON_KEYS,      FBCON_VALUES},
  {0, 2, "shadow",       FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "fbcon",        FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "dummy",        FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "em85xxyuv",    FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "em85xxosd",    FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "svpxxosd",     FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "utpmc",        FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "bf533",        FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "mb93493",      FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "commlcd",      FBCON_KEYS,      FBCON_VALUES},
  {0, 1, "dfb",          FBCON_KEYS,      FBCON_VALUES},
  {0, TABLESIZE(SYSTEMFONT_KEYS), "systemfont",  SYSTEMFONT_KEYS, SYSTEMFONT_VALUES},
  {0, 1, "cursorinfo",   CURSORINFO_KEYS, CURSORINFO_VALUES},
  {0, 1, "iconinfo",     ICONINFO_KEYS,   ICONINFO_VALUES},
  {0, 1, "bitmapinfo",   BITMAPINFO_KEYS, BITMAPINFO_VALUES},
/* optional sections */
  {0, 2, "bgpicture", BGPICTURE_KEYS, BGPICTURE_VALUES},
  /*
  {1, "mouse", MOUSE_KEYS, MOUSE_VALUES},
  */
  {0, 2, "event", EVENT_KEYS, EVENT_VALUES},
#ifdef _IME_GB2312
  {0, 3, "imeinfo", IMEINFO_KEYS, IMEINFO_VALUES},
#endif
#if defined (_TTF_SUPPORT) || defined (_FT2_SUPPORT)
  {0, TABLESIZE(TTFINFO_KEYS), "truetypefonts", TTFINFO_KEYS, TTFINFO_VALUES},
#endif
};

static ETC_S MGETC = {0, TABLESIZE (mgetc_sections), mgetc_sections};

GHANDLE __mg_get_mgetc (void)
{
    return (GHANDLE) &MGETC;
}

#endif /* !DYNAMIC_LOAD */
#endif /* _INCORE_RES */
```

3.3 the Sample of Configuration

Under most circumstances, we modify runtime configuration file, we will be limited to several sections. The system section and font related several sections are primary sections. In this chapter, we will give two configuration examples.

3.3.1 Runtime Configuration when only Support for ISO8859-1 Charset

1) Configuration File

```
# The first system font must be a logical font using RBF device font.
[systemfont]
font_number=1
font0=rbf-fixed-rrncnn-8-16-ISO8859-1

default=0
wchar_def=0
fixed=0
caption=0
menu=0
control=0

[rawbitmapfonts]
font_number=1
name0=rbf-fixed-rrncnn-8-16-ISO8859-1
fontfile0=/usr/local/lib/minigui/res/font/8x16-iso8859-1.bin

[varbitmapfonts]
font_number=0
```

```
[qpf]
font_number=0

[truetypefonts]
font_number=0

[type1fonts]
font_number=0
```

2) Incore Configuration Options

```
static char *SYSTEMFONT_KEYS[] =
{"font_number", "font0", "default", "wchar_def", "fixed", "caption", "menu", "control"};

static char *SYSTEMFONT_VALUES[] =
{
    "1","rbf-fixed-rrncnn-8-16-ISO8859-1", "0", "0", "0", "0", "0", "0"
};
```

3.3.2 Specifying Different Graphic Engine and Input Engine

1) Configuration File

```
[system]
# GAL engine and default options
gal_engine=commlcd

# IAL engine
ial_engine=auto

mdev=/dev/ts
mtype=IMPS2
```

2) Incore Configuration Option

```
static char *SYSTEM_KEYS[] = {"gal_engine", "ial_engine", "mdev", "mtype"};

static char *SYSTEM_VALUES[] = {"commlcd", "auto", "/dev/ts", "IMPS2"};
```

4 Developing MiniGUI Application in Windows

Feynman provides two methods for developer, which is accustomed to develop application in Window platform.

- Using the package of MiniGUI for Win32. It is pre-compiled standard development package in Win32. It contains wvfb, MiniGUI function library (libminigui and libmgext) and header files.
- Using MiniGUI SDK for Win32[4]. This is an optional component in MiniGUI-VAR. It contains the whole source codes and provides users the convenience for customizing the package of MiniGUI for Win32.

By using the package of MiniGUI for Win32 or the component product of MiniGUI SDK for Win32, developer can compile and debug MiniGUI application in Windows.

This chapter describes how to use the package of MiniGUI for Win32. User can contact Feynman to purchase the component product of MiniGUI SDK for Win32.

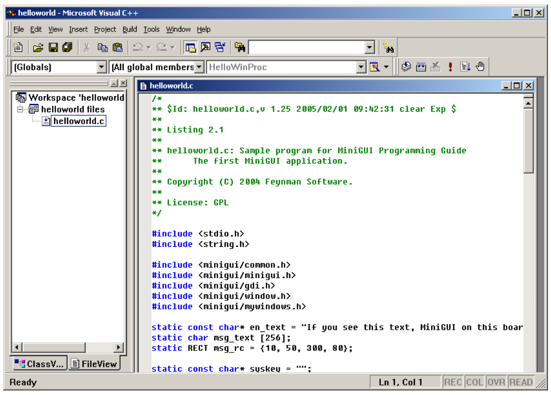
The package of MiniGUI for Win32 is located in the directory win32-dev of cdrom, and filename is **`minigui-dev-1.6.10-win32.tar.gz`**.You can visit Feynman software web and download it from the following web address.

```
http://www.minigui.com/download/cindex.shtml
```

To develop MiniGUI application in Windows, you must install MS Visual Studio 98. First, you decompress arbitrary directory in windows. Secondly you open the helloworld project file in VC according to README. Figure 4.1 shows it.

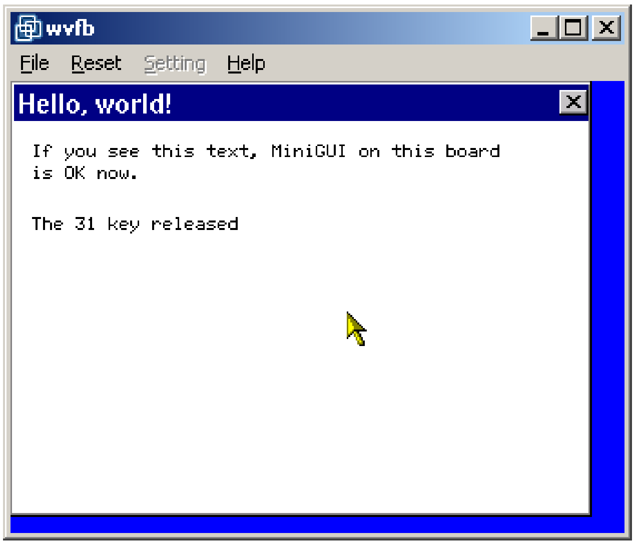
After compiling successfully, you should run wvfb first and run helloworld. Note that you need copy **`helloworld.ext`** to directory dll. Fig 4.2 shows running result.

[4] MiniGUI SDK for Win32 only support MiniGUI-Threads and in-core mode. Because of the limitation of platform, you can’t use the font support for TrueType and Type1 and load JPEG and PNG, which need the support of third-library.



```
/*
** $Id: helloworld.c,v 1.25 2005/02/01 09:42:31 clear Exp $
**
** Listing 2.1
**
** helloworld.c: Sample program for MiniGUI Programming Guide
**      The first MiniGUI application.
**
** Copyright (C) 2004 Feynman Software.
**
** License: GPL
*/

#include <stdio.h>
#include <string.h>

#include <minigui/common.h>
#include <minigui/minigui.h>
#include <minigui/gdi.h>
#include <minigui/window.h>
#include <minigui/mywindows.h>

static const char* en_text = "If you see this text, MiniGUI on this boar
static char msg_text [256];
static RECT msg_rc = {10, 50, 300, 80};

static const char* syskey = "";
```

Fig 4.1 open MiniGUI helloworld project



```
If you see this text, MiniGUI on this board
is OK now.

The 31 key released
```

Fig 4.2 Compiling and Running MiniGUI Application in Windows

Refer to above helloworld, you can create, develop and compile new MiniGUI application in VC. But give your attention to the following:

- Because the package of MiniGUI for Win32 is pre-compiled library, the function, compiling configuration options, and running configuration options are fixed, and only support MiniGUI-Threads runtime mode.
- Using the package of MiniGUI for Win32 to develop applications, please don’t call Windows special API, which isn’t supported possibility by target OS.

Appendix A Frequent Ask Questions (FAQs)

A.1 Questions Relevant to GPL Versions

Q1. Do I need to pay Feynman Software for the license fee if I use GPL versions of MiniGUI?

A1. The GPL versions of MiniGUI are available at Feynman Software website; you can use them freely if you use MiniGUI under GPL license. However, the release of your applications that are based on MiniGUI GPL versions should also complies with GPL. If you use MiniGUI to develop commercial purpose applications, i.e., you do not want to release them under GPL terms, you then should pay Feynman Software for the licensing fee.

Q2. When you use MiniGUI GPL versions, what kind of behaviors would violate Feynman Software's legal rights?

A2. Feynman Software owns the copyright of several free software projects. We release that software under GPL with the purpose of helping users to understand software inner mechanism well and customize them freely and easily. However, most users are not familiar with GPL terms; they would sometimes act against GPL terms unconsciously. The behaviors below would violate Feynman Software's legal rights:

- Pirate part or whole source code to use in other occasions; the worse thing is to pirate MiniGUI and sell it as private software. Such behavior has already seriously offended against the copyright laws.
- Modify source code of free software, and use them in commercial purpose, but they are not released according to GPL terms.

Under GPL terms, applications based-on MiniGUI should be released under GPL. If you do not release MiniGUI applications under GPL, neither buying MiniGUI commercial licenses, this behavior belongs to software pirate.

A.2 Questions Relevant to MiniGUI Application Fields

Q3. What kinds of products that use MiniGUI are successfully launched in market?

A3. MiniGUI is widely used in the products like mobile phones, IPTVs, digital TVs, industry control systems, information terminals, industrial meters, and so on. For the detailed introduction for some typical products, you can visit:

```
http://www.minigui.com/project/index.shtml.
```

Q4. How is about the stability of MiniGUI?

A5. It is hard to answer this question as the factor that influences system stability is sometimes due to applications instead of the libraries. However, we can offer you some information as reference:

- For a complicated MiniGUI application, a test shows that there is no problem for the shift in between multi-windows by pressing key 100,000 times in two days.
- Many industrial control systems that are developed based on MiniGUI can now stably run under real industrial situations.

A.3 Questions Relevant to Portability

Q5. What operating systems does MiniGUI support?

A5. By now, MiniGUI provides the support for many popular embedded operating systems including Linux/uClinux, VxWorks, ThreadX, Nucleus, pSOS, OSE, eCos, and even uC/OS-II. MiniGUI can also run on Win32 platform.

Q6. Which CPUs have MiniGUI run on successfully so far? Moreover, what is the lowest frequency of CPU MiniGUI needed?

A6. There are successful cases for MiniGUI running in ARM-based CPUs (such as StongARM, xScale, S3C2410, S3C2440, EM8511, EM8620), PowerPC, MIPS, M68k, FRV.

In those CPUs, the one with lowest main frequency is about 20 MHz (20 MIPS).

Q7. Would MiniGUI provide support for monochrome LCD?

A7. Yes. Actually, MiniGUI can provide support for almost all LCD controllers in various modes, such as monochrome, gray, 256-color, 4096-color, and 65536-color.

Q8. Which resolution of screen can MiniGUI run properly?

A8. In theory, the running of MiniGUI is not influenced by the resolution of screen.

A.4 Questions Relevant to Compilation

Q9. Why are there so many compilation errors when I enable the option to support TrueType font?

A9. The main reason is that the libttf version supporting TrueType font in your system is too high. MiniGUI uses libttf 1.3.1. In several Linux distributions such as RedHat Linux 7, the library libttf 2.0 is installed by the default. In this case, you can install libttf 1.3.1 or use **`--disable-ttfsupport`** option to disable the support for TrueType font of MiniGUI.

Q10. During compiling the library, why does the mistake below occur sometimes?

```
    can not make hard link filename.o to filename.lo.
```

A10. Symbol links and hard links are the specialized file types in UNIX file system. If you compile library being maintained by Automake/Autoconf script, you cannot create these links on a non-UNIX file system. Please check your file system to make sure if it is not FAT32 file system.

Q11. When I use the Open File Dialog Box, why does the mistake below occur?

```
undefined reference to ShowOpenDialog
```

A11. The function ShowOpenDialog is included in the MiniGUI extended library. If you want to use this function, you should include two header files: <**minigui/mgext.h**> and <**minigui/filedlg.h**>. When make the executable, please make sure to link libmgext (**-lmgext**). In addition, if you run MiniGUI on some embedded operating systems, which are lack of the support for file system, you can't use the Open File Dialog Box.

Q12. My system does not support 64-bit integer. Is the data type of Uint64 in MiniGUI essential?

A12. The data type of Uint64 in MiniGUI is used to generate the complex graphics. If your system does not support 64-bit integer, you can use the following configuration option to disable the usage of 64-bit integer:

```
--disable-fixedmath
```

A.5 Questions Relevant to Input Engines

Q13. On Linux PC boxes, what kinds of mouse types does MiniGUI support?

A13. Currently, the mouse protocols supported by MiniGUI are MS, MS3, PS2, and Intelligent PS2 (IMPS2).

Q14. On Linux PC boxes, I would like to use the old serials mouse. What should I do?

A14. MiniGUI can provide support for almost all mouse types via GPM. Please configure it as follows:

1) Run **gpm -k** to kill gpm that is running.
2) Run **mouse-test** to confirm your mouse device and protocol.
3) Run **gpm** to set mouse device and protocol as follows.

```
gpm -R -t <yourmousetype> -m <yourmousedevice>
```

4) Edit **MiniGUI.cfg** file, set **mtype** as **gpm**; and set mdev as **/dev/gpmdata**:

```
[system]
...
```

```
mtype=gpm
mdev=/dev/gpmdata
```

Then, start up MiniGUI. Please note you can use the option `-R` when you set the mouse protocol by `gpm`. `-R` option is used to transfer original mouse protocol to GPM defined mouse protocol, and make it shown in `/dev/gpmdata` file.

A.6 Runtime Questions

Q15. On Linux PC boxes, how would I close the input bar after starting up MDE program?

A15. You can use the left `<Ctrl>` key to switch input bar when MDE starts up; additionally, you can also configure `imenumber` option in `MiniGUI.cfg` file. Set `imenumber` as `imenumber = 0`, then the input bar would not be shown when MDE starts up.

```
[imeinfo]
imenumber=0
```

Q16. On Linux, How would I capture the screen of MiniGUI?

A16. When running MiniGUI program, you can capture the screen as a BMP file in the current directory by pressing `<PrtSc>` key. The file name is `0-<NO>.bmp`, therein `<NO>` is the number of times of pressing `<PrtSc>` key. You can save the BMP file of the current active main window as `<HWND>-<NO>.bmp`, therein `<HWND>` is the handle of the active main window while `<NO>` is the number of times of pressing `<Ctrl+PrtSc>` key.

Q17. Why does the program exit after displaying two dialog boxes when I run `mginit` in MDE?

A17. The main reason is that MiniGUI being installed does not provide support for PNG image files. In some Linux distributions (such as early TurboLinux), as the version of their PNG graphics support library (libpng) is too old, it would automatically disable the support for PNG image when you configure MiniGUI. In this case, `LoadBitmapFromFile` function of MiniGUI cannot correctly load PNG image files, while MDE `mginit` needs to load two PNG files for running. That is why `mginit` exits.

To solve this problem, there are two ways. First, you can download and install the latest libpng library from INTERNET. Secondly, modify **`nr`** value in section **`[mginit]`** in **`mginit.rc`**, and make the value less than 8.

Another reason that may cause such error is that you do not start up **`mginit`** in its directory. Please change to the directory, then run **`mginit`**.

Q18. Under MiniGUI-Processes or MiniGUI-Lite runtime mode, how would I switch from MiniGUI to other console?

A18. Under MiniGUI-Processes or MiniGUI-Lite runtime mode, if you are using the **`console`** input engine, you can switch from MiniGUI to other virtual console by pressing **`<Right_Ctrl+Fx>`** key, also, you can quit MiniGUI by pressing **`<Ctrl+Alt+Backspace>`**. Currently, MiniGUI-Threads does not provide such functions.

A.7 Common Error Messages

Q19. Why is the following message shown when I run programs in MDE on Linux?

```
AttachSharedResource: No such file or directory
Error in step 6: Can not attach shared resource!
Initialize minigui failure when using /etc/MiniGUI.cfg as cfg file.
```

A19. If you configure MiniGUI as MiniGUI-Processes or MiniGUI-Lite, you should run **`mginit`** program first. As MiniGUI-Processes or MiniGUI-Lite adopts a C/S architecture, you have to start up the sever program, **`mginit,`** before running client programs. In MDE package, you should run **`mginit`** in **`mginit/`** directory first, then run demo programs in other directories.

Q20. Why do I see the information below when I run MiniGUI?

```
GAL ENGINE: Error when opening /dev/fb0: Permission denied. Please check your kernel config.
GAL: Init GAL engine failure.
Error in step 3: Can not initialize graphics engine!
Initialize minigui failure when using /usr/local/etc/MiniGUI.cfg as cfg file
```

A20. The main reason is that you have not activated the FrameBuffer driver yet, or the permission of **`/dev/fb0`** is incorrect.

Q21. Under MiniGUI-Processes or MiniGUI-Lite runtime mode, why does it give error information below when I run mginit in MDE?

```
    Error in step 2 : There is already an instance of minigui.
    Initialize minigui failure when using /usr/local/etc/MiniGUI.cfg as config file.
```

A21. Usually, there are two possible reasons. One is that you have already run an **`mginit`** program; other is that you did not exit MiniGUI properly when you run **`mginit`** last time. If it is the second reason, you can delete **`minigui`** file and **`mginit`** file in **`/var/tmp/`** directory. If it still does not work, please restart your computer.

Q22. Why do the following statement show when I run MiniGUI?

```
NEWGAL: Does not find matched engine: fbcon.
Error in step 3: Can not get graphics engine information!
```

A22. The possible problem is that **`FBCON`** engine in NEWGAL interface fails when initializing FrameBuffer device. The main reasons are that your kernel does not support FrameBuffer driver, or does not activate FrameBuffer driver, or you have no proper access permission to open **`/dev/fb0`** device.

Q23. On Linux, what is the meaning of the error information below?

```
vesafb does not support changing the video mode
```

A23. It is a warning that can be ignored. It aims at VESA FrameBuffer driver. VESA FrameBuffer driver does not support the display mode switch during running. It can only set video mode by the boot option for kernel. Moreover, once set, it cannot be changed unless you modify the boot option and restart your system.

Q24. On Linux, what is the meaning of the error information below?

```
NEWGAL: No video mode large enough for the resolution specified.
```

```
NewGAL: Set video mode failure.
```

A24. The main reason is that the display resolution being set in **`MiniGUI.cfg`** is higher than that supported by your FrameBuffer driver. Therefore, you can try to set a smaller resolution by modify **`MiniGUI.cfg`** file.

Appendix B Index