MiniGUI-Lite 服务器程序精简版

（陈云川 ybc2084@163.com UESTC,CD 2007-04-15）

1 引言

对于MiniGUI-Lite而言，其运行方式为客户端/服务器模式。通常，MiniGUI-Lite的服务器程序被称为mginit。MiniGUI的综合演示程序（MDE）中包含的mginit显得太过于复杂了，更要命的问题是运行时还会出现段错误。因此，作者决定将其改写为下面这样一个最精简的mginit服务器程序，首先给出程序源代码，然后再对程序中的关键部分进行说明。有关开发定制的MiniGUI-Lite服务器程序mginit的详细说明请参考《MiniGUI编程指南for MiniGUI Ver 1.3.x》（参考文献[1]，可从www.minigui.com下载）。

```
/*
 * $file   :        mginit.c
 * $desc   :        simplest MiniGUI server
 * $author     :    rockins(ybc2084@163.com)
 * $date   :        Sun Apr 15 17:42:58 CST 2007
 * $copyright :     all copyrights(c) reserved by rockins.
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <signal.h>
#include <time.h>
#include <sys/types.h>
#include <sys/wait.h>

#include <minigui/common.h>
#include <minigui/minigui.h>
#include <minigui/gdi.h>
#include <minigui/window.h>
#include <minigui/control.h>
#include <minigui/mgext.h>

int
MiniGUIMain (int args, const char* arg[])
{
    MSG msg;
```

```
    OnNewDelClient = NULL;
    OnChangeLayer = NULL;

#if _MINIGUI_VERSION_CODE < _VERSION_CODE (1, 3, 0)
    if (!ServerStartup ()) {
        fprintf (stderr, "Can not start the server of MiniGUI-Lite: mginit.\n");
        return 1;
    }
#endif

    if (SetDesktopRect (650, 490, 660, 500) == 0) {
        fprintf (stderr, "Empty desktop rect.\n");
        return 1;
    }

    if (!InitMiniGUIExt ()) {
        fprintf (stderr, "Can not init mgext library.\n");
        return 1;
    }

    while (GetMessage (&msg, HWND_DESKTOP)) {
        DispatchMessage (&msg);
    }

    MiniGUIExtCleanUp ();

    return 0;
}
```

2 运行效果

采用如下命令编译这个 mginit 程序：

```
[root@cyc mginit]# arm-linux-gcc -o mginit mginit.c \
> -I/home/cyc/minigui/libminigui-1.3.3/arm-linux-build/include \
> -I/home/cyc/minigui/tslib/arm-linux-build/include \
> -L/home/cyc/minigui/libminigui-1.3.3/arm-linux-build/lib \
> -L/home/cyc/minigui/tslib/arm-linux-build/lib \
> -lmgext -lminigui –lts
```

由于程序要和 minigui 库链接，而 minigui 库又要和 tslib 库连接。因此指定了动态链接

库的顺序：

```
-lmgext -lminigui –lts
```

注意：上述链接顺序不能改变，其原因很简单，mgext 是建立在 minigui 的基础上的，而 minigui 又要用到 tslib，所以必须以上述顺序进行链接。如果列位看官对为什么要连接 tslib 感到疑惑的话，请参阅在下之前的两篇帖子。

此外，两个-I 选项指定的是 minigui 头文件的位置和 tslib 的头文件所在的位置。而两个 -L 选项指定的是 minigui 和 tslib 的动态链接库所在的位置。

最后将编译生成的服务器程序 mginit 通过 NFS 下载到目标板上，运行后看到的效果如图 1 所示。

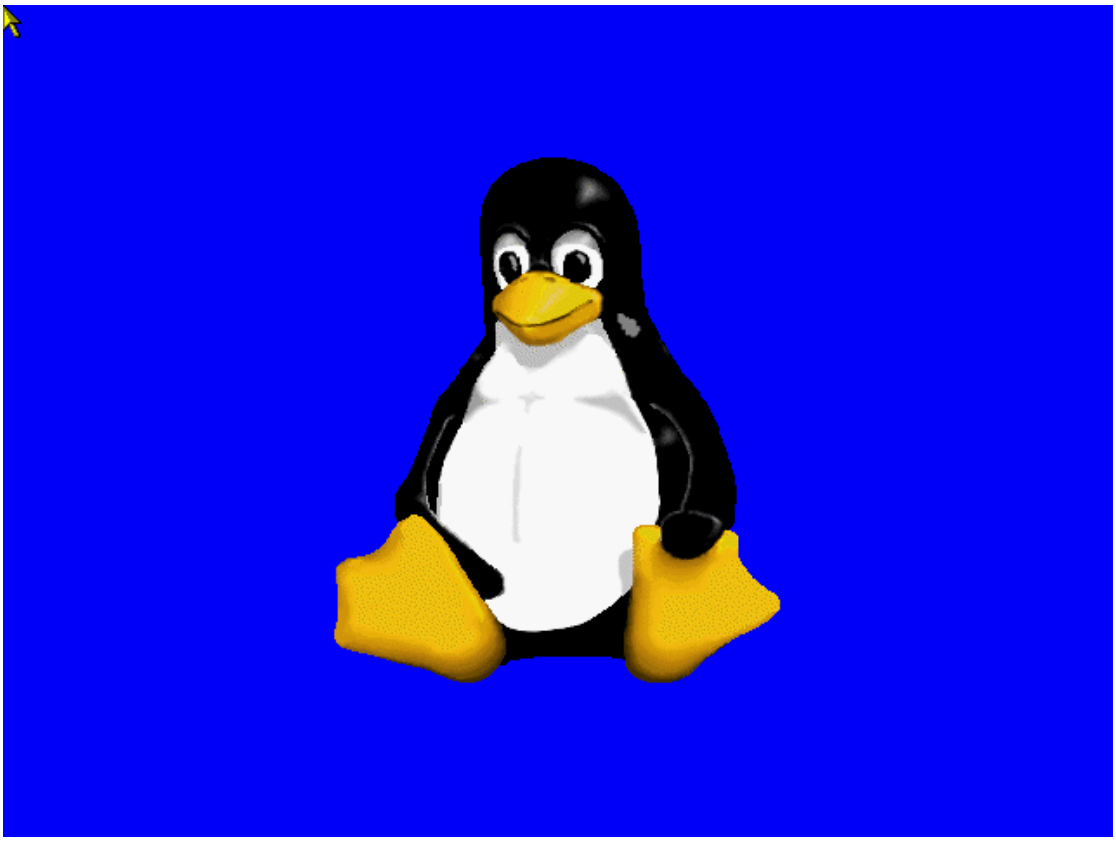

图 1 MiniGUI-Lite 的服务器程序 mginit 截图

3 关键点解说

无论是服务器程序还是客户端程序，MiniGUI 程序的入口点都是 MiniGUIMain()。上文中，在 MiniGUIMain()函数中定义了一个 MiniGUI 消息结构 msg，在消息循环中，从窗口消息队列获取的消息将存放到此消息结构中并将其交给相应的窗口过程函数去处理。

OnNewDelClient 和 OnChangeLayer 是 MiniGUI 定义的两个全局函数指针，它们只能用于 MiniGUI 服务器中。可以将它们指向定制的处理函数，当有客户端与 mginit 服务器建立连接或者断开连接的时候就将调用 OnNewDelClient 所指的函数，而当层（Layer）发生改变的时候就将调用 OnChangeLayer 所指的函数。由于在本示例中不需要关心客户端的变化和

层的变化，因此将这两个函数指针都置为 NULL。

ServerStartup()函数的功能是启动服务器，其完成的工作主要是打开一个 Unix 域套接口/var/tmp/minigui 并在其上监听来自 MiniGUI 客户端的连接。如果 ServerStartup()执行成功，将返回 TRUE，否则返回 FALSE 表示失败。注意：在采用 MiniGUI-Lite 时，系统中只能有一个 MiniGUI 服务器——也即这里的 mginit，如果同时启动多个 mginit，则只有第一次启动的 mginit 会成功，其它都将失败。

SetDesktopRect()函数的作用是设定由 mginit 程序独占的桌面区域。这里设定的区域为(650, 490, 660, 500)。各位看官可能会对此感到奇怪，因为在前面的两篇拙文中，已经提到 Sitsang 评估板的液晶显示屏的分辨率大小为 640x480，显然，这里为 mginit 设置的独占区域已经在屏幕可见的范围之外了。实际上，背后的原因是这样的：对于 MiniGUI 客户端程序而言，只能在服务器 mginit 独占的区域之外进行绘制。由于作者打算把整个屏幕都留给客户端程序使用，因此就不能让 mginit 占用屏幕上的可见区域，而是指定了一个屏幕范围之外的区域，这个区域的具体位置并不重要，只要在屏幕可见范围之外即可。MiniGUI 允许服务器独占的桌面区域位于可见的屏幕区域之外。另外，任何时候客户端程序都不能在服务器独占的桌面区域上进行绘制输出，而服务器程序尽管设定了桌面区域，却可以在任何时候在屏幕的任何位置进行绘制输出。显然，服务器 mginit 的权力比客户端程序的权力要大。

InitMiniGUIExt()函数初始化了 mgext 库，这个库对应的动态共享库文件为 libmgext-1.3.so.3.0.0，它提供了对 MiniGUI 核心库 libminigui-1.3.so.3.0.0 的扩展支持。主要是提供了一些扩展控件和皮肤界面的支持。如果要使用这些扩展空间或者皮肤界面，必须在调用任何 mgext 库的 API 函数之前调用 InitMiniGUIExt()完成相关的初始化工作。当 InitMiniGUIExt()函数执行成功时，返回 TRUE，否则返回 FALSE。由于本例中并未使用到 mgext 库的功能，因此这里这是为了起到演示的目的。

在 MiniGUIMain()函数快要结束的地方实现了整个服务器的消息循环。在这个循环中，GetMessage()函数首先从桌面窗口的消息队列中取出消息。桌面窗口 HWND_DESKTOP 是所有窗口的祖先窗口。只要取出的消息不为 MSG_QUIT，那么 GetMessage()就将把消息复制到 msg 中，并且 GetMessage()函数将会返回非零值，同时复制到 msg 消息结构中的消息将被 DispatchMessage()分发给对应的主窗口，由该主窗口的窗口过程函数进行处理；如果 GetMessage()发现取出的消息为 MSG_QUIT，那么 GetMessage()函数将返回 0。这将导致 mginit 退出消息循环。在 DispatchMessage()中，不需要再指明消息要分发到的主窗口，因为 MSG 消息结构中已经包含了消息要发往的主窗口的句柄（handle）。

最后，MiniGUIExtCleanUp()函数的作用是清理 mgext 库占用的资源。当不再需要使用任何 mgext 的 API 函数时，应该调用此函数。

4 总结

可以看到，实现一个最最简单的 MiniGUI-Lite 服务器程序实际上只需要几十行代码就够了。“麻雀虽小，五脏俱全”。

5 参考文献

[1] 北京飞漫软件技术有限公司。MiniGUI 编程指南 for MiniGUI Ver 1.3.x。2003 年 10 月

[2] 北京飞漫软件技术有限公司。MiniGUI 用户手册 for MiniGUI Ver 1.3.x。2003 年 10 月

[3] 北京飞漫软件技术有限公司。MiniGUI API Reference Documentation for MiniGUI Ver 1.3.x。2003 年 10 月