

# 嵌入式系统

## An Introduction to Embedded System

### 第十二课、嵌入式系统的GUI

教师：蔡铭

cm@zju.edu.cn

浙江大学计算机学院人工智能研究所  
航天科技—浙江大学基础软件研发中心

# 课程大纲

 图形用户界面及嵌入式GUI简介

 嵌入式Linux典型GUI系统简介

 嵌入式GUI实现关键技术

# 用户界面发展历史（1/2）

## □ 什么是用户界面（UI）？

定义：计算机用户界面是指计算机与其使用者之间的对话接口，是计算机系统的重要组成部分。

## □ 用户界面发展历史：

- ✓ 早期的计算机通过面板上的**指示灯**显示二进制数据和指令，人们通过面板上的**开关、扳键及穿孔纸带**送入各种数据和命令。
- ✓ 50年代中、后期，采用**作业控制语言(JCL)**及**控制台打字机**，使计算机批处理多个计算任务，代替原来笨拙的手工扳键方式。

# 用户界面发展历史（2/2）

## □ 用户界面发展历史：

- ✓ 1963年，美国MIT在709/7090计算机上开发出第一个分时系统CTSS，并最早使用了文本编辑程序。从此，以**命令行形式**对话的多用户分时终端成为70年代~80年代用户界面的主流。
- ✓ 80年代初，由美国Xerox公司Alto计算机首先使用的Smalltalk—80程序设计开发环境，以及Apple的Lisa、Macintosh等计算机，将用户界面推向**图形用户界面（GUI）**的新阶段。

图形用户界面（GUI）的广泛流行是当今计算机技术的重大成就之一，它极大地方便了非专业用户的使用，人们不再需要死记硬背大量的命令，而可以通过窗口、菜单方便地进行操作。

# 图形用户界面的特征（1/2）



## □ WIMP

- ✓ W(Windows): 指窗口，是用户的工作区域。一个屏幕上可以有多个窗口。
- ✓ I(Icons): 指图标，是一种形象化的图形标志，易于人们隐喻和理解。
- ✓ M(Menus): 指菜单，可供用户选择的功能提示。
- ✓ P(Pointing Devices): 指鼠标器等，便于用户直接对屏幕对象操作。

## □ 用户模型

- ✓ GUI采用了不少桌面办公的隐喻，使应用者共享一个直观的界面框架。由于人们熟悉办公桌的情况，因而，对计算机显示的图符含义容易理解，诸如：桌面、文件夹、收件箱、画笔、工作簿、时钟等。

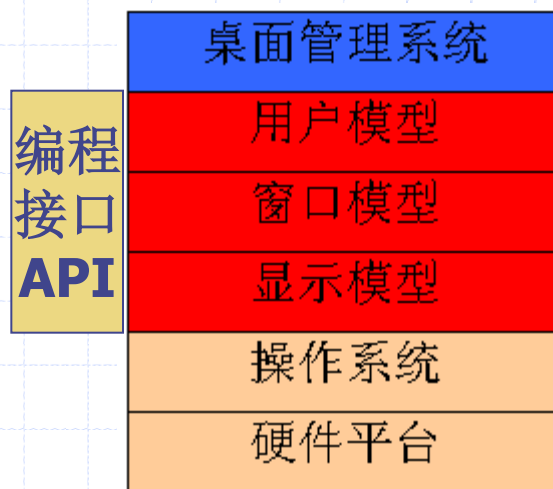
# 图形用户界面的特征（2/2）

## □ 直接操作

- ✓ 命令行的用户界面需要记忆大量命令，而且需要指定操作对象的位置，如：行号、空格数、X及Y的坐标等。
- ✓ 采用GUI后，用户可直接对屏幕上对象进行操作，如：拖动、删除、插入、放大和旋转等。
- ✓ 用户执行操作后，屏幕能立即给出反馈信息或结果，实现了“所见即所得”（What You See Is What You Get）。



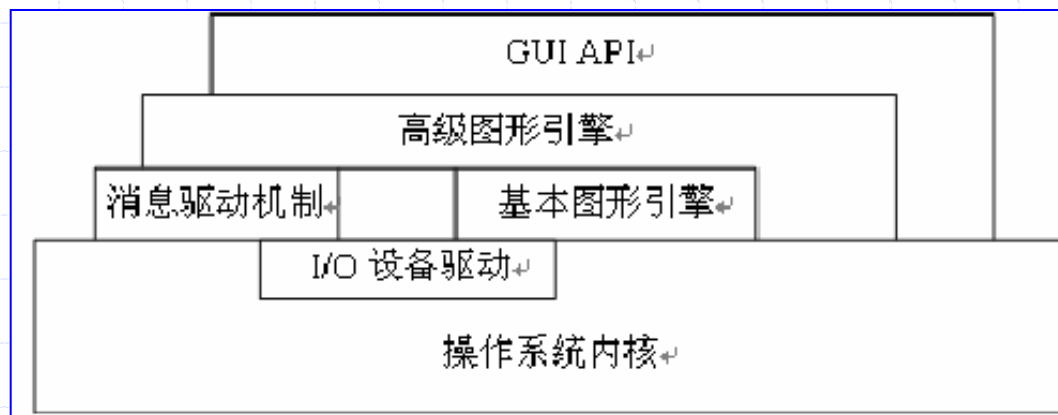
# 图形用户界面系统的结构模型



## □ 一个图形用户界面系统通常由三个基本层次组成：

- ✓ 显示模型：决定了图形在屏幕上的基本显示方式，如：UNIX上的X窗口显示模型；MS Windows的图形设备接口(GDI)等。
- ✓ 窗口模型：确定了窗口如何在屏幕上显示，如何改变大小、如何移动，及窗口的层次关系等。
- ✓ 用户模型：又称为图形用户界面的视感，是指如何在屏幕上组织各种图形对象，以及这些对象之间如何交互。

# 图形用户界面系统的功能模型



□ 一个图形用户界面系统的功能模型主要包括：

- ✓ I/O设备驱动
- ✓ 消息驱动管理
- ✓ 基本图形引擎
- ✓ 高级图形引擎
- ✓ 应用编程接口



# 用户界面系统的发展趋势

- 以用户为中心
- 多通道 (Multimodality)
- 智能化非精确交互技术
- 高带宽信息输入
- 图示编程 (Visual Programming) 支持



# 典型GUI系统 — X窗口系统简介（1/4）

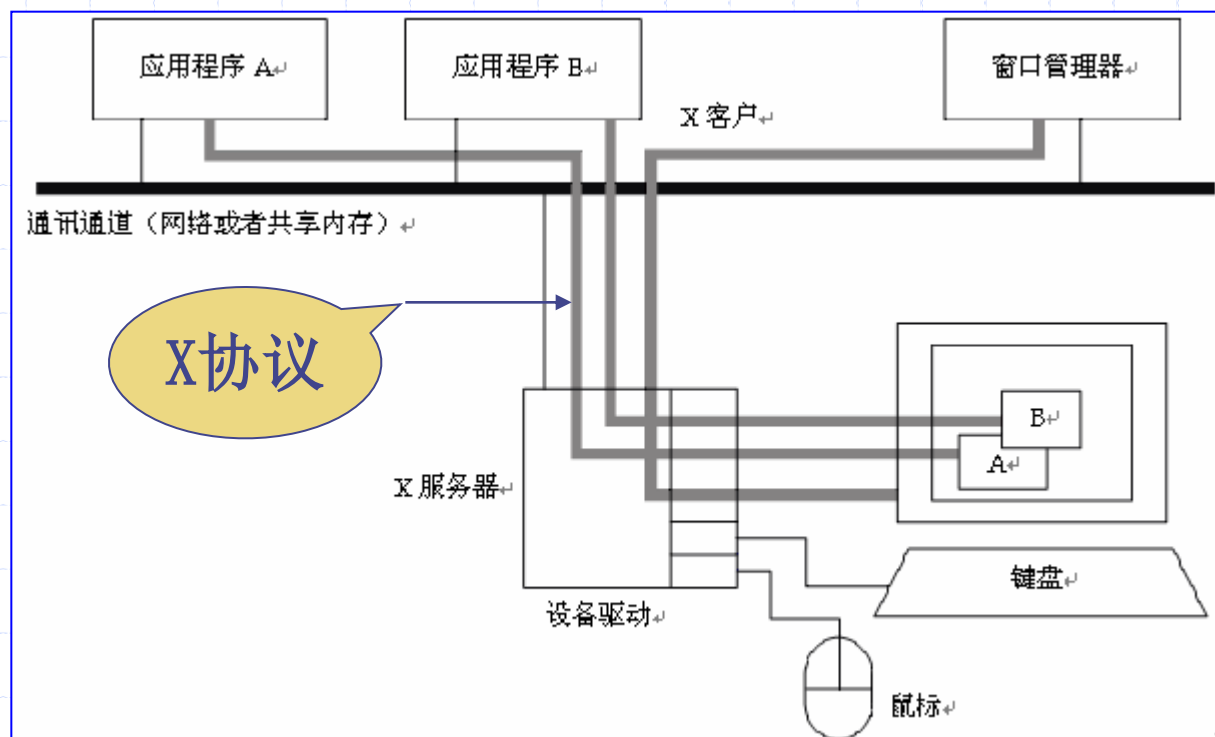
- X窗口系统(X-Window System, 简称X窗口)是Unix/Linux上标准的图形界面, 虽然各类Unix上的图形界面不完全相同, 但其图形界面都以X协议为基础。
- 1983年, MIT为了发展校园网络, 发起了一个名为Athena的计划, X窗口就是Athena计划的产物, 也特别地溶入了网络的概念。
- X窗口原名为W窗口, 取名自Window, 后经过改良的W窗口以英文字母排序中的下一个字母为名, 称为X窗口。

# 典型GUI系统 — X窗口系统简介（2/4）

## □ XFree86计划:

- ✓ 早期X窗口是构建在Unix工作站上，没有PC版。
- ✓ XFree86计划成立的目的是提供一个PC版的X窗口。
- ✓ XFree86虽然不是以GPL授权，但是它也可以自由拷贝、散播，也可以使用在商业用途上，所以大部份的PC 版Unix，如Linux、BSD等都将XFree86加入在操作系统的套件内。

# 典型GUI系统 — X窗口系统简介（3/4）

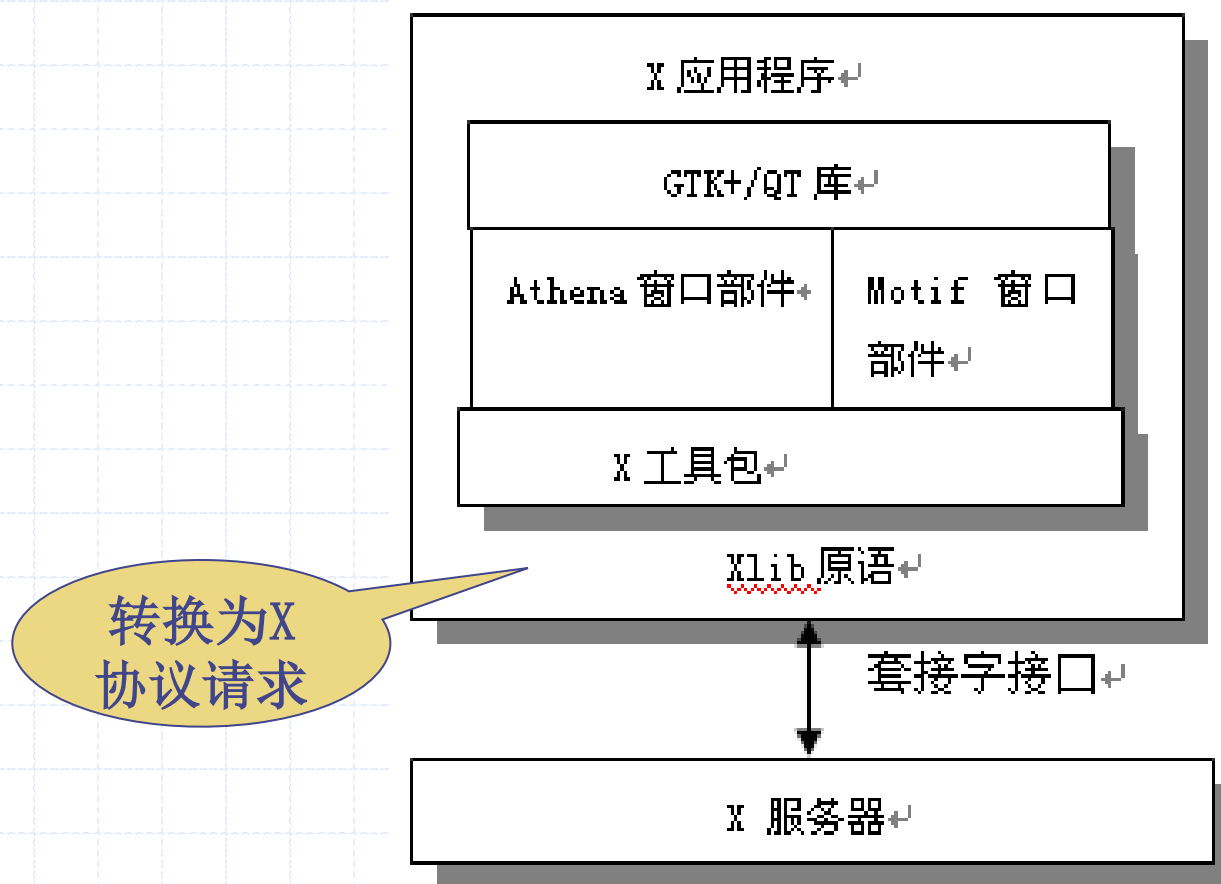


## □ X窗口系统架构：

- ✓ 采用客户端/服务器的设计概念
- ✓ 由3部分组成：客户端、服务器和X协议
- ✓ 事件驱动机制

# 典型GUI系统 — X窗口系统简介（4/4）

## □ X窗口函数库



# 嵌入式GUI特点

□ 嵌入式**GUI**要求简单、直观、可靠、占用资源少，且反应快速，以适应系统硬件资源有限的条件。具体特点如下：

- ✓ 体积小
- ✓ 占用系统资源少
- ✓ 可靠性高
- ✓ 上层接口与硬件无关，高度可移植
- ✓ 在某些应用场合应具备实时性

# 课程大纲

 图形用户界面及嵌入式GUI简介

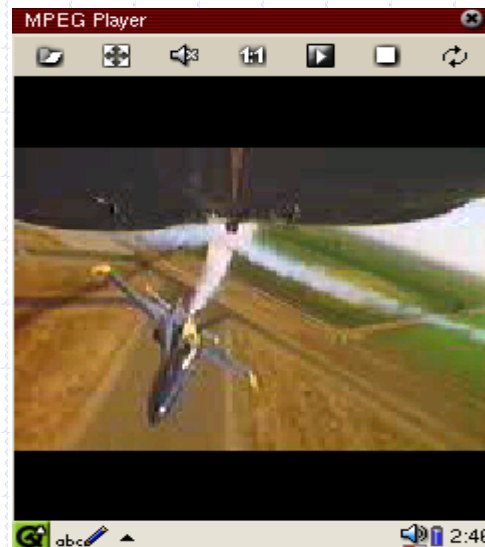
 嵌入式Linux典型GUI系统简介

 嵌入式GUI实现关键技术

# 嵌入式Linux典型GUI—QT/Embedded

□ QT/Embedded是著名的Qt库开发商挪威Trolltech公司开发(被Nokia收购)的基于C++、跨平台、嵌入式GUI

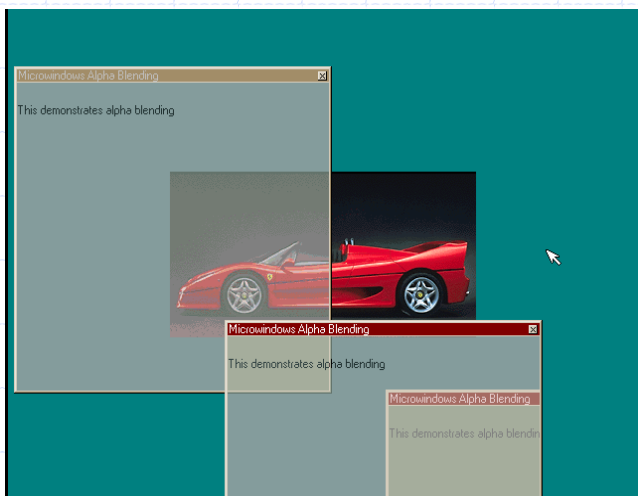
- ✓ 主要针对手持式信息终端
- ✓ 有商业版本和开源版本（**LGPL**）
- ✓ 完全面向对象，拥有良好的扩展性与稳定性
- ✓ 提供丰富的类库，包含大量的可重用类
- ✓ 移植性好
- ✓ 代码规模大
- ✓ 实时性较弱





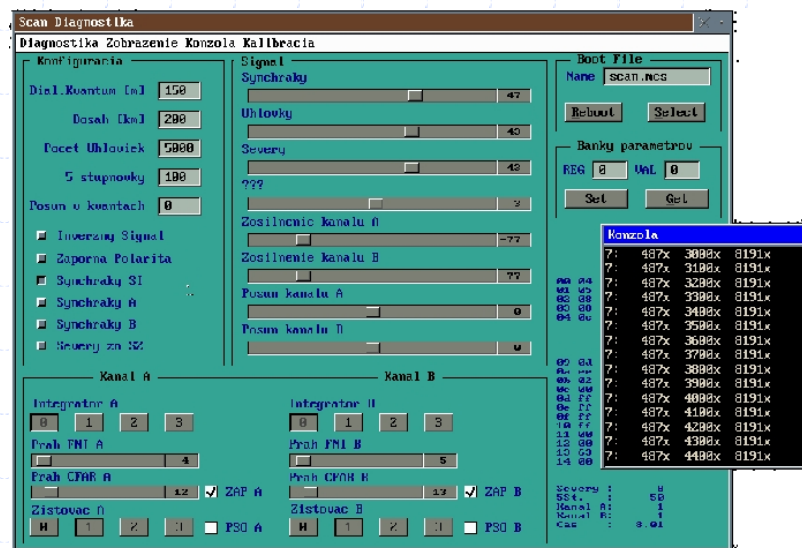
# 嵌入式Linux典型GUI—MicroWindows/NanoX

- MicroWindows是一款面向嵌入式Linux系统的GUI，由美国Century software公司主持开发，用于工控机、机顶盒等。
  - ✓ 开放源代码（GPL）
  - ✓ 支持Framebuffer、SVGALib库等进行图形显示
  - ✓ 提供了Alpha 混合、三维支持、TrueType字体
  - ✓ 基于客户端/服务器体系结构
  - ✓ 可移植性好



# 嵌入式Linux典型GUI—OpenGUI

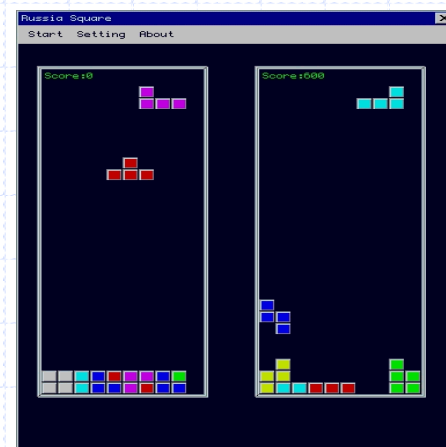
- ❑ OpenGUI最早称为FastGL，支持多种操作系统平台，如MS-DOS、QNX、Linux，主要用于开发图形应用程序、游戏。
  - ✓ 开放源代码（GPL）
  - ✓ 基于汇编级的图形内核，并利用MMX指令优化，运行效率高
  - ✓ 在Linux上基于Framebuffer、SVGALib实现绘图，支持Mesa3D
  - ✓ 适合于X86平台
  - ✓ 可移植性较差
  - ✓ 系统运行稳定、可靠



# 嵌入式Linux典型GUI—MiniGUI

□ **MiniGUI**是由北京飞漫软件公司开发的一个跨平台、嵌入式GUI，主要应用于工业控制、STB、手持设备，支持Linux/uClinux、VxWorks、threadX、Nucleus等os。

- ✓ 开放源代码（GPL）
- ✓ 提供了完备的多窗口机制
- ✓ 多字符集和多字体支持
- ✓ 全拼、五笔等汉字输入
- ✓ 系统结构小巧



魏永明:从MiniGUI看嵌入式十年收获与失去 (2012-04-10 15:49:43)

标签：杂谈

魏永明:从MiniGUI看嵌入式十年收获与失去

来源：电子工程世界网 发布者：电子工程世界网

时间：2012年4月10日 06:58

## 飞漫软件十年回顾




发表于：2012年04月06日 19:26

北京飞漫软件技术有限公司（飞漫软件）成立于2002年，今年是第十个年头了。嵌入式软件技术在中国的发展历程。本文将回顾飞漫软件的十年历程。回味过去，些启迪。

	MiniGUI	MicroWindows	OpenGUI	QT/Embedded
API	Win32风格	X、Win32风格	私有	QT (C++)
API是否完备	是	Win32不完善	是	是
函数库典型大小	500K	600K	300K	1.5M
可移植性	很好	很好	只支持X86平台	较好（函数库跨平台交叉编译困难）
授权条款	GPL/商业许可证	MPL/LGPL	LGPL	QPL/GPL/商业许可证
多进程支持	好	X支持好，Win32不支持	不好	好
健壮性/稳定性	好	很差	一般	差
多语种支持	独特的多字符集支持功能	一般	一般	UNICODE，效率低

	MiniGUI	MicroWindows	OpenGUI	QT/Embedded
可配置和可定制性	好，大量编译配置选项	一般	差	差
系统资源消耗	小	较大（基于UNIX套接字，进程间通讯）	最小（不支持多进程）	最大（C++）
效率	好	较差	最好	差
操作系统支持	Linux/uClinux，uC/OSII，VxWorks等	Linux	Dos、Linux、QNX	Linux
硬件平台支持	X86、ARM、MIPS、PowerPC	X86、ARM、MIPS	X86	X86、ARM
主要应用区域	中国大陆、台湾地区	美国，及国内少数用户	欧洲	欧美、韩国

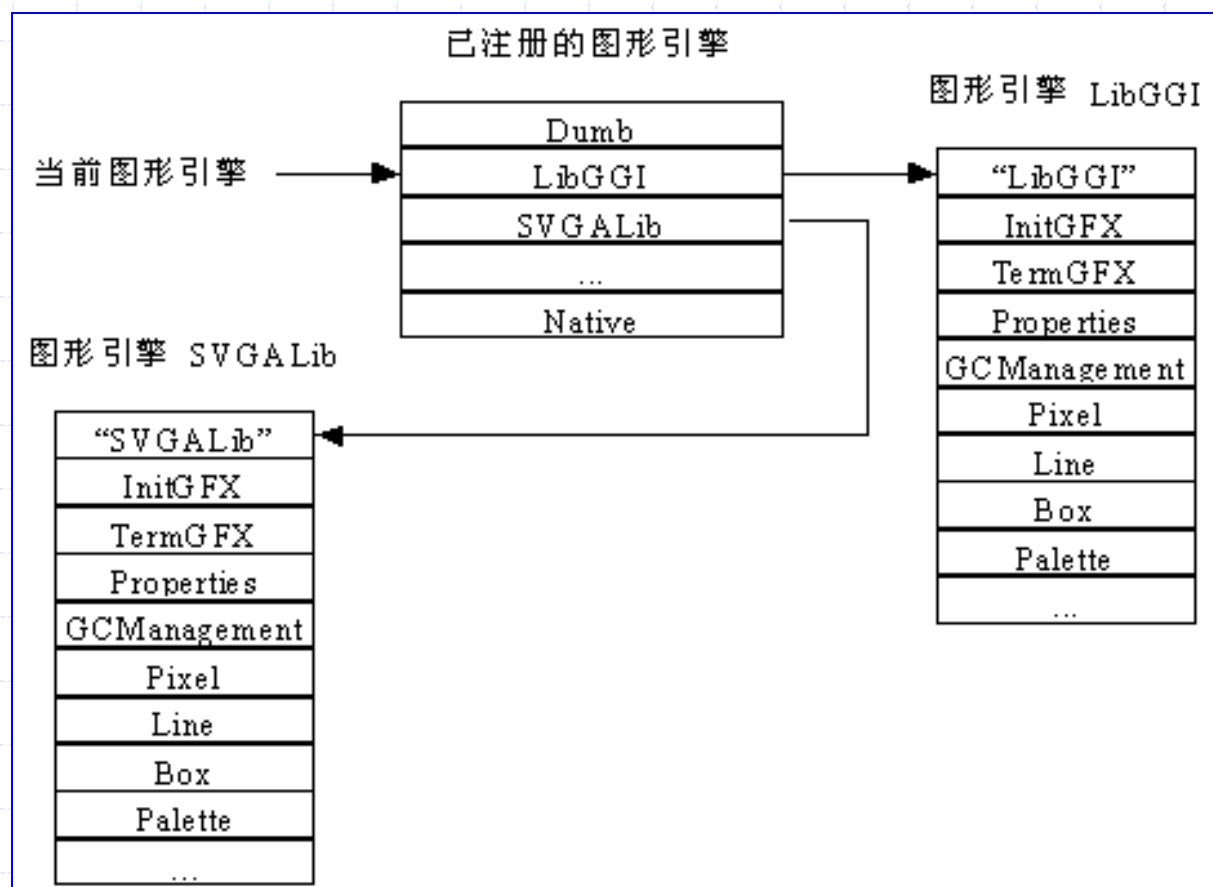
# 课程大纲

-  图形用户界面及嵌入式GUI简介
-  嵌入式Linux典型GUI系统简介
-  嵌入式GUI实现关键技术

# 基于Linux的基础图形引擎选择

- 基础图形引擎负责完成基本图元绘制、窗口剪切等功能，作为其他高级图形功能的基本函数库。基于Linux的引擎包括：
  - ✓ **X Window:** Xlib、X Intrinsic、Toolkits (Tiny-X是XServer在嵌入式系统的实现)
  - ✓ **SVGA Lib:** 支持标准的VGA 图形模式和一些其他的模式
  - ✓ **LibGGI:** 采用共享库模式，提供异步绘制模式
  - ✓ **FrameBuffer:** 在Linux2.2.x内核中实现的一种驱动程序接口，将显示设备抽象为帧缓冲区。

# 多图形引擎支持





# 客户机/服务器模式

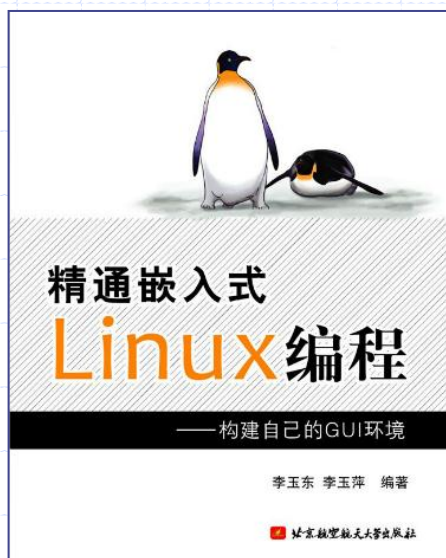


- ❑ 多个任务可以并发发送图形显示要求，为避免屏幕区域竞争，**GUI**系统往往采用客户/服务模式，由服务器负责输出仲裁。
- ❑ 客户/服务模式的两种交互模型：
  - ✓ 服务器负责图形绘制：**X Window**
  - ✓ 服务器负责提供绘制区域，客户负责图形绘制：**LGUI**

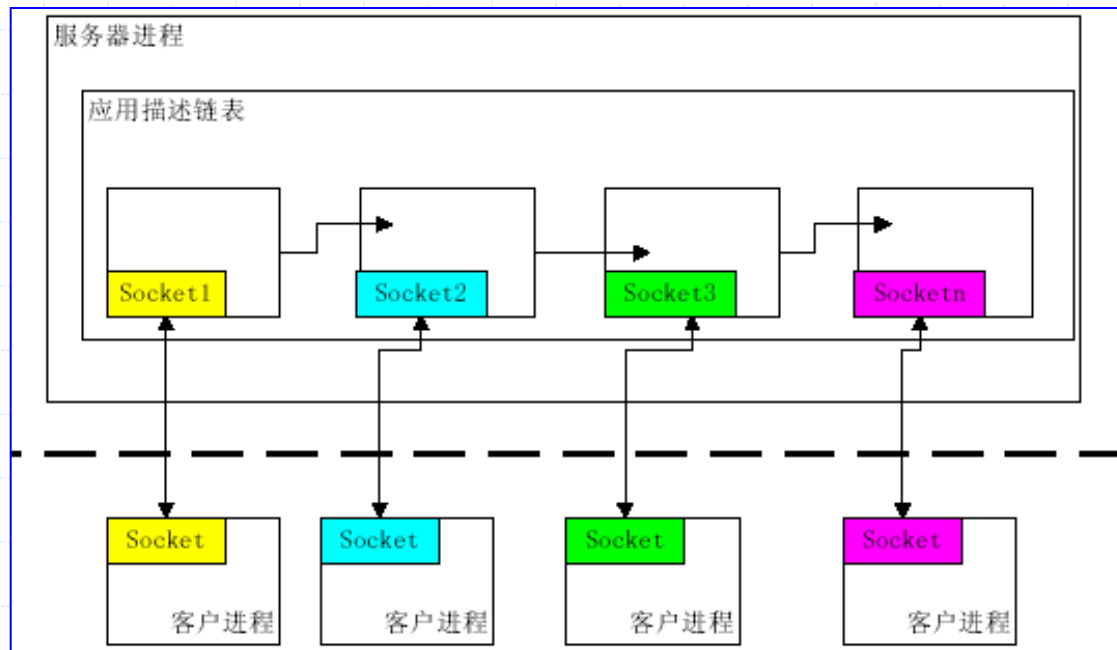
# 嵌入式Linux典型GUI—LGUI

□ **LGUI**是中国开发的另一个基于**Linux**、开源的轻量级嵌入式**GUI**，由深圳龙硅科技主持开发负责开发。

- ✓ 开放源代码（GPL）
- ✓ 基于**Linux FrameBuffer**的**GDI**图形引擎
- ✓ 多进程、多线程支持的微型客户机/服务器模式系统
- ✓ 多窗口与剪切域的管理、消息队列的管理

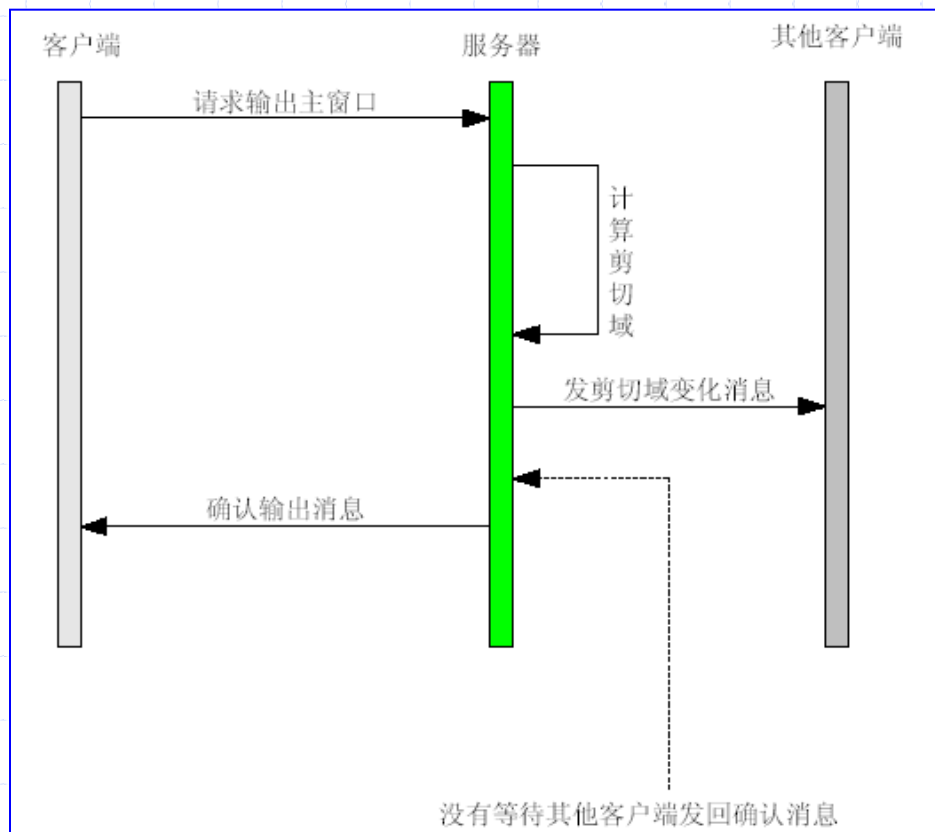


# 基于socket的客户/服务模式—LGUI

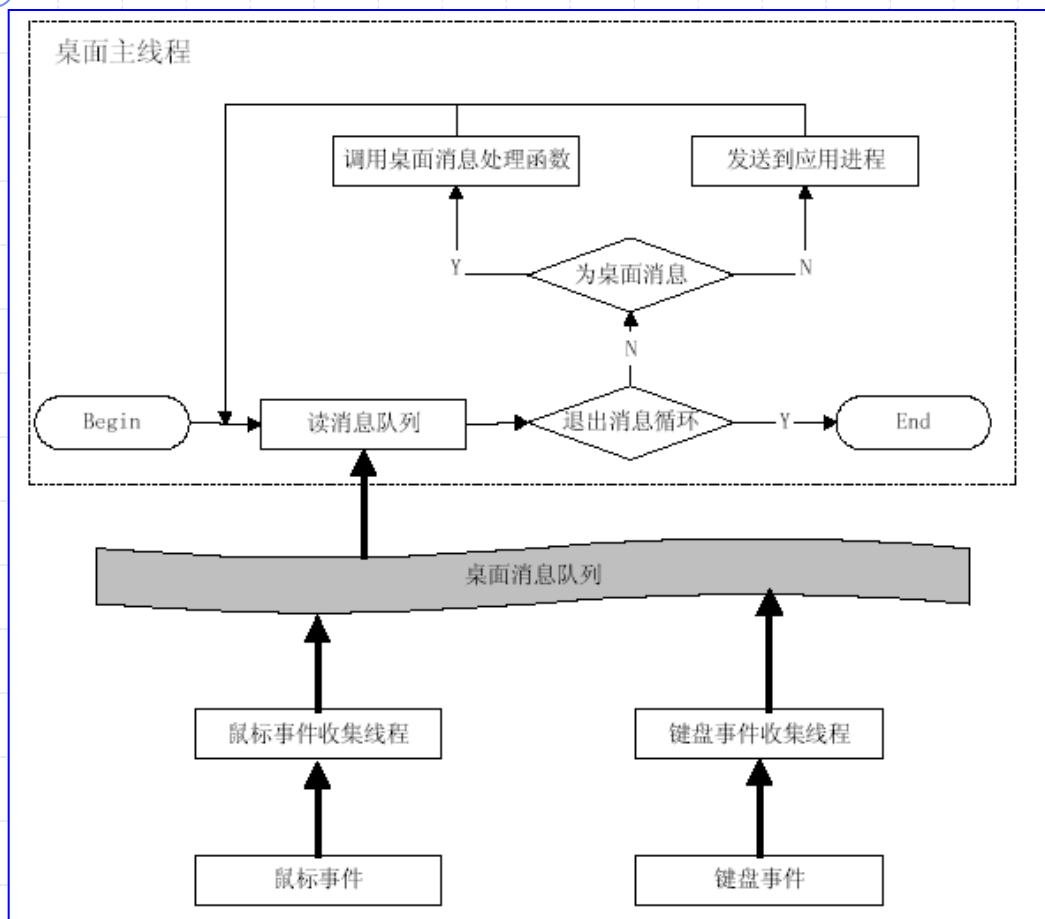


- ✓ 客户与服务器之间的连接通过**Socket**来实现。
- ✓ 在服务器端，对每一个客户端连接请求都创建一个**Socket**进行通讯。
- ✓ 客户正常退出或异常终止，服务器进程都会收到一个关闭**Socket**的消息，服务器可以清理对应的客户资源。

# 客户/服务交互模式—LGUI

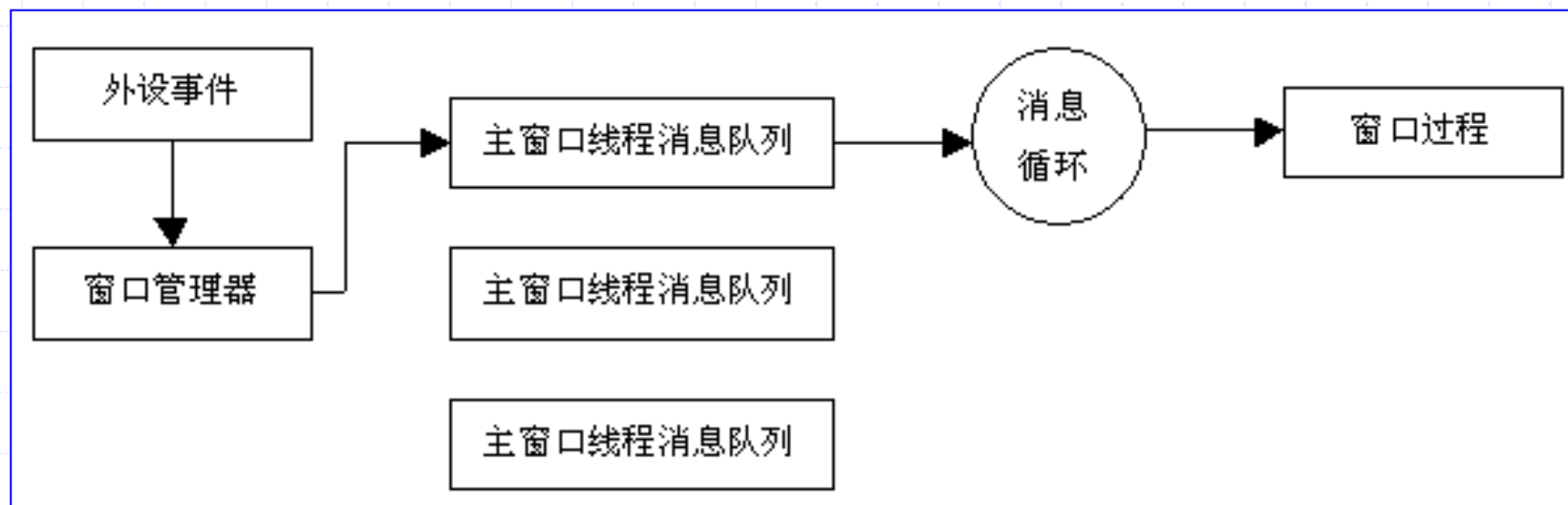


# 外部消息收集与分发—LGUI

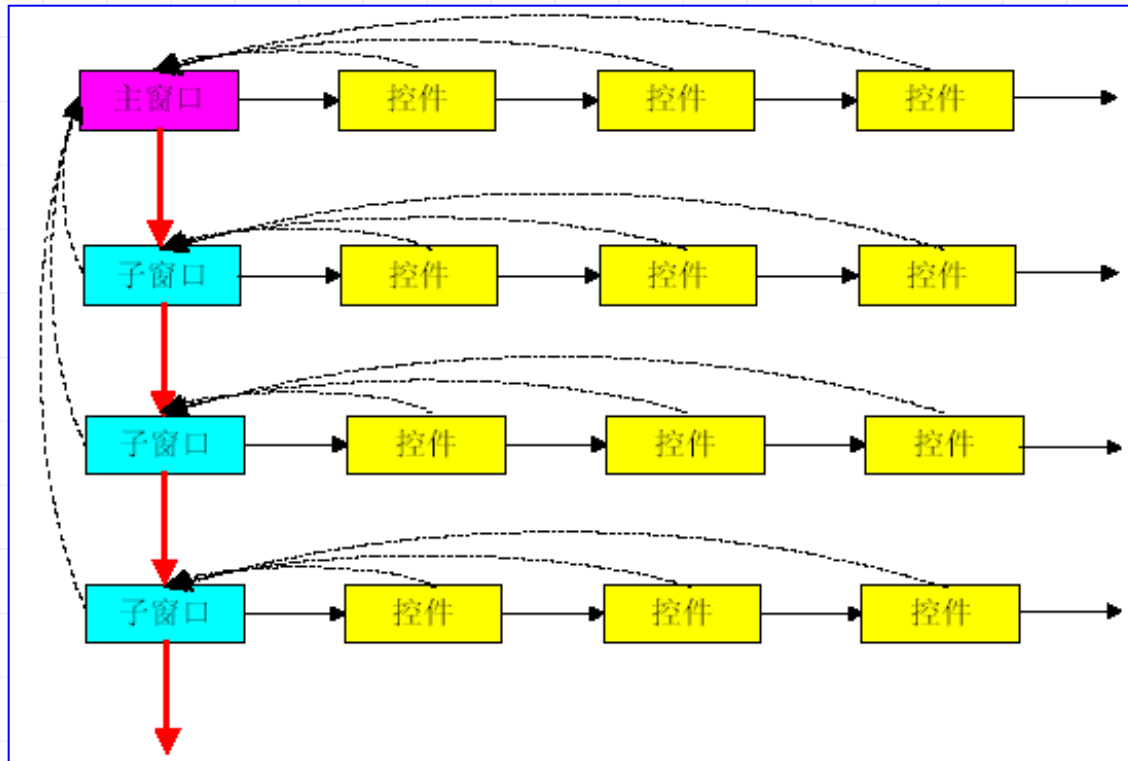


- 外部消息由桌面进程处理，还是转发到应用进程中去，主要根据：
  - ✓ 当前鼠标位置
  - ✓ 主窗口当前激活状态
- 应用进程接收到消息后，需进一步存入窗口消息队列，找到相应的接收控件进行处理。

# 消息驱动的多进程/多线程机制—LGUI



# 多级窗口树管理—LGUI



- ✓ 1个主窗口可以含有多个子窗口
- ✓ 1个子窗口可以含有多个控件
- ✓ 主窗口上的子窗口以儿子/兄弟方式链接
- ✓ 所有子窗口/控件都有指向父窗口的指针

# GUI系统实现若干关键算法（1/6）

## □ 消息处理

- ✓ 多窗口环境下，多进程/多线程消息数据传递
- ✓ 消息传递机制：同步消息、异步消息
- ✓ 消息优先级处理

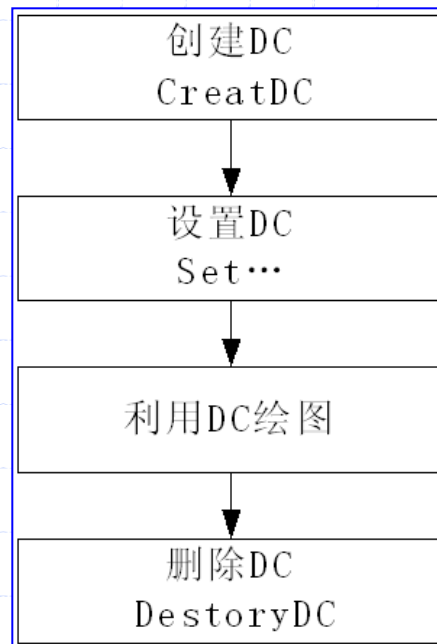


# GUI系统实现若干关键算法（2/6）

## □ 图形设备上下文（DC）

- ✓ 图形设备上下文（DC）是绘图的关键。
- ✓ DC保存了每一个绘图对象的相关参数，可以保证多任务环境中，不同任务的绘图参数相互独立。

```
typedef struct {  
    int DrawPointx;  
    int DrawPointy;    //绘图所使用的坐标点  
    int PenWidth;      //画笔宽度  
    U32 PenMode;       //画笔模式  
    U32 PenColor;      //画笔的颜色  
    int DrawOrgx;      //绘图的坐标原点位置  
    int DrawOrgy;  
    int DrawRangex;    //绘图的区域范围  
    int DrawRangey;  
    U8 bUpdateBuffer;  //是否更新后台缓冲区  
    U32 Fontcolor;     //字符颜色  
} DC, *PDC;
```

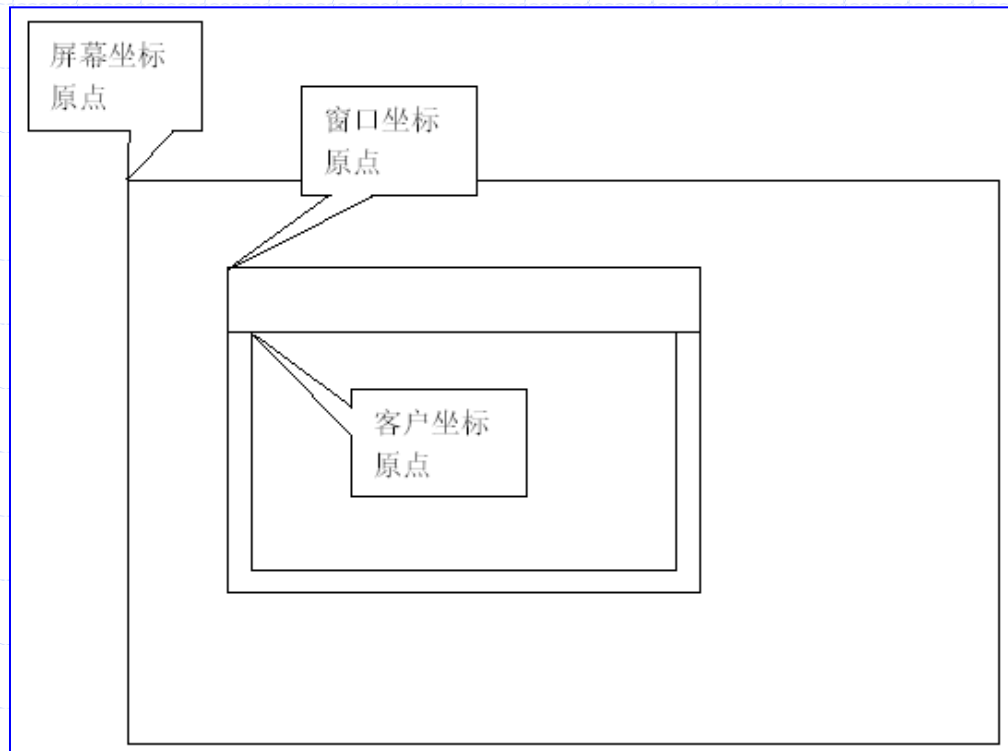


- ✓ 在调用图形输出函数时，要求指定经初始化，或经建立的图形设备上下文，或设备环境。

# GUI系统实现若干关键算法（3/6）

□ 映射模式：指定了特定图形输出的坐标值如何映射到图形设备的坐标值

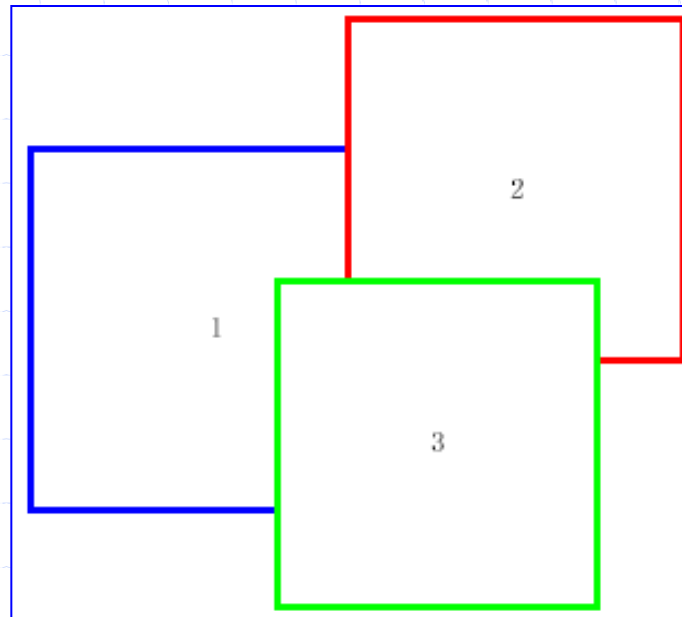
✓ 常用的图形设备坐标如下：



# GUI系统实现若干关键算法（4/6）

## □ 窗口管理与Z序

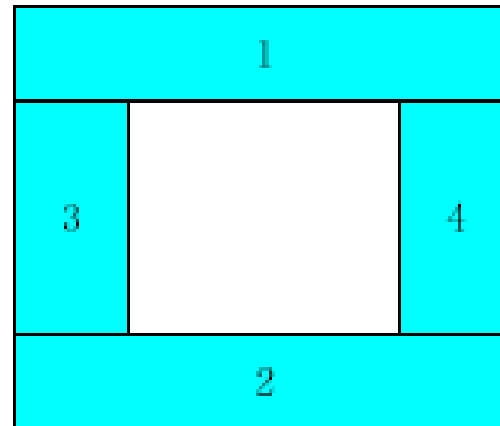
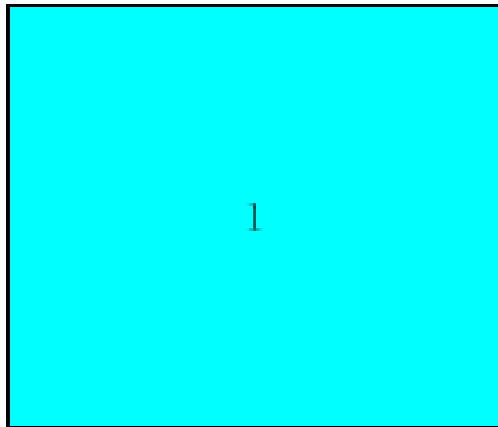
- ✓ 窗口的**Z序**是窗口在屏幕上体现出来的前后关系，**Z序**越大，则对应的窗口越靠近上层。
- ✓ **Z序**大的窗口将会剪切**Z序**值比它小的所有窗口。



# GUI系统实现若干关键算法（5/6）

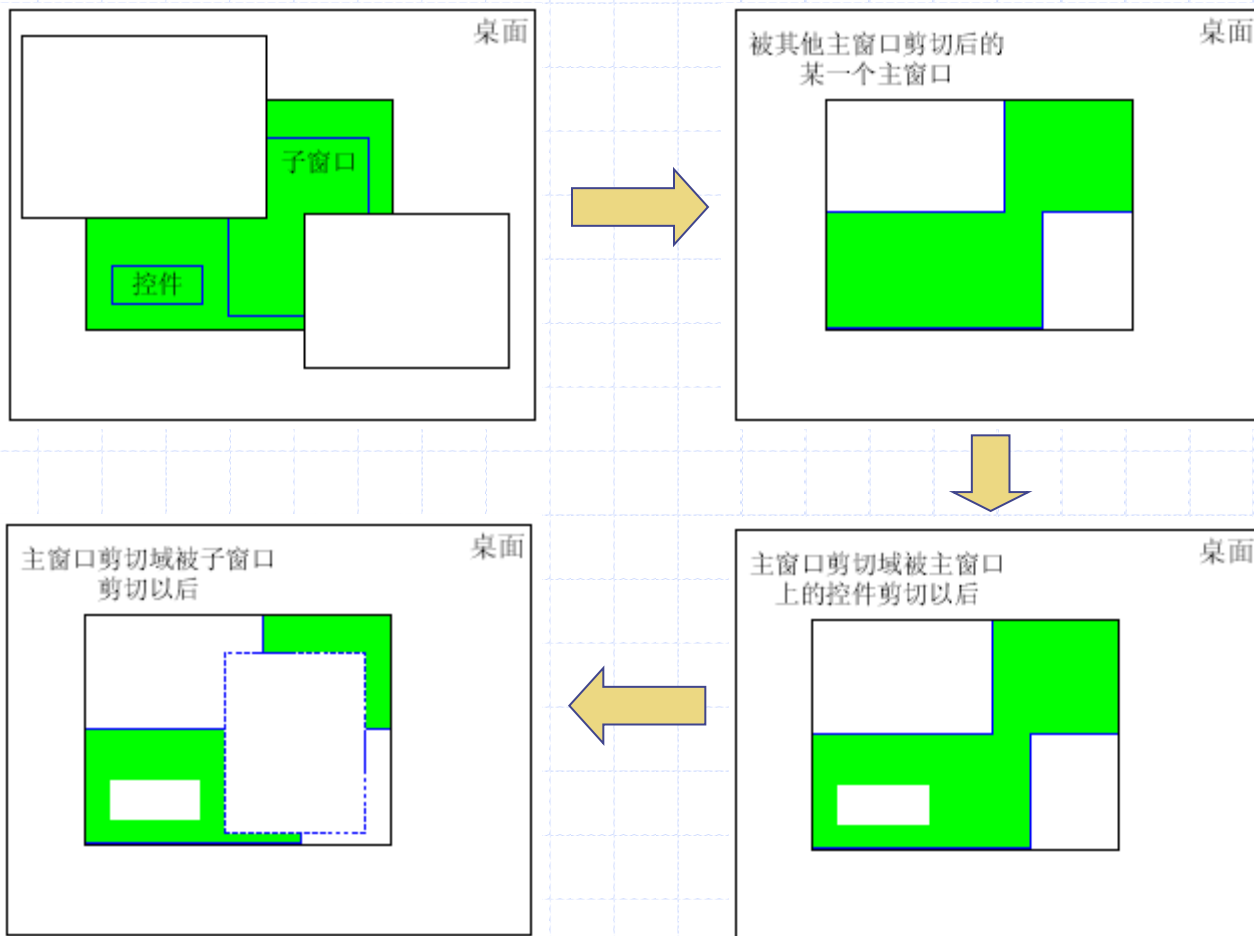
## □ 剪切算法

- ✓ 由于多窗口具有频繁的互相覆盖情况，而产生相互剪切，因而需要构造一个高效的剪切域维护算法。
- ✓ 窗口剪切域生成：剪切矩形



# GUI系统实现若干关键算法 (6/6)

## □ 主窗口剪切算法





谢谢!

