基于 Tilcon 航海标绘台界面设计

高贤志　乔立伟　张崇猛　侯巍

（天津航海仪器研究所　天津　300131）

基金名称：综合航桥系统及关键设备，课题编号：2011AA110201

摘　要：针对基于 WINDML 的航海标绘台系统中编程代码量大、界面显示粗糙等特点，结合航海标台系统应用要求，采用 Tilcon 软件设计了系统操控流程和应用界面，降低了软件编码的工作量，增强了可靠性和可维护性，解决了在 VxWorks 下 Tilcon 汉字显示等关键问题。应用结果表明，该方法开发的图形界面可靠性好、实时性强、界面效果丰富，具有广泛的实用性和推广价值。

关键词：Tilcon　VxWorks　图形开发　程序设计　航海　标绘台

0　引言

航海标绘台是舰船上航海作业的重要设备，对于嵌入式系统来说，一个拥有图形界面的应用程序，人机界面的简洁性和美观性往往是考虑的重要指标。在目前的嵌入式 VxWorks 图形界面开发工具中，主要有三种方式：WindML3.0 + Tornado2.2，不用现成的界面开发工具，完全在 WindML 下，自己编写代码，调用媒体库实现图形界面，开发难度很大；WindML2.0.3 + Tornado2.2 + Zinc6.0（for Tornado2.2），Zinc6.0 作用同 Tilcon IDS，但由于其不稳定性，或者说是会有冲突，功能上相对不够完善，经常会出现一些意想不到的问题，现已被风河公司放弃，WindML3.0 已经不兼容 Zinc6.0 了；WindML3.0 + Tornado2.2 + Tilcon IDS5.5，Tilcon 是目前最先进的 Vxworks 下实时操作系统图形开发工具，同步支持最新版本的 Tornado/ VxWorks 以及 WindML 多媒体库。Tilcon 采用了最先进的图形技术，不会有任何用户自己编写的程序与图形系统混合，并且不会影响系统的安全性和稳定性。基本解决了困扰嵌入式实时图形领域应用的难题，具有极高的可靠性和可维护性。

1　Tilcon 的开发模式

从用户的角度，使用 Tilcon 开发用户接口的过程应该是：进入集成开发环境 IDS 后，通过拖拽工具条上的用户接口对象创建 Screen，并保存为 twd 文档。右键单击打开属性窗口编辑对象。所有的变化都将立即反映到用户创建的工作窗口。与此同时，用户还可以看见一系列函数和事件管理产生的变化。用户可以通过点击开发环境的测试按钮来查看图形界面的运行效果。用户的 C/C++ 应用程序需要包含主事件循环，控制 Tilcon 的用户接口。应用程序使用 TRT_Start 命令启动 EVE（完全独立于你的应用的一个进程），同时打开一个通信通道。当执行初始化命令后，应用程序将使用 API 命令，让引擎知道如何加载和显示拖拽方式开发的窗口文件。启动加载和显示窗口之后，应用程序将进入主循环，等待事件的发生。如果这时发生一个 GUI 事件（例如单击按钮）事件可以由 Tilcon 引擎直接处理，也可以由 Tilcon 引擎发送一个数据结构给用户应用程序。这样，这个描述事件就可以根据用户的意愿由应用程序代码处理。也可以由回调函数（callback）处理事件。控制进程（用户应用程序）将发送命令（API 命令）到一个队列中，等待引擎处理。命令是异步的，控制进程无需等待画面更新。引擎把 GUI 调用和控制进程分离出来。它是一个独立进程，保证了控制进程的实时性，提高了系统的可靠性。一旦用户编辑好了 Screen，控制进程写好，控制进程与 Tilcon API 进行链接与编译，就可以运行了。

2　VxWorks 下图形开发原理

在 Tornado 开发环境下，首先根据自己的应用和目标机的需要，对 VxWorks 模块进行配置，包括：C++ 模块、图形模块、目标机硬件模块、操作系统

模块等 其次选择 VxWorks 镜像类型，构造 VxWorks 镜像，最后引导 Vx - Works 镜像，生成目标机上所需的 VxWorks 镜像文件。我们还需要对 WindML 进行配置和编译以及 Tilcon 的裁减与配置，然后进行图形界面的应用程序开发。在 Tilcon Interfaee Builder 开发环境下实现图形界面的设计，包括拖拉所需的控件及各个控件的驱动代码，然后生成图形库文件，再根据 Tornado 与 Tilcon 之间的无缝链接关系，将 Tilcon 生成的图形库文件加载到 Tornado 环境中，从而实现嵌入式下的图形开发。它们的层次关系如图 1 所示。

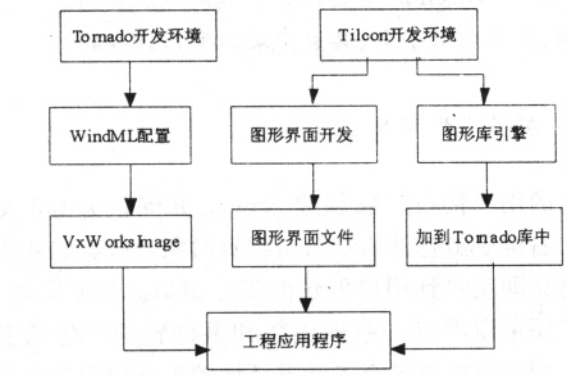

图 1 嵌入式图形开发模式

用户使用工具拖拽控件放置到主程序窗口内并设置各控件的显示和运行属性。用户开发的应用图形界面完成后，存储为数据资源文件，用户的应用程序 (APSOURCE) 与 Tilcon 提供的图形引擎、API 编译链接在一起最终形成产品。数据资源文件的独立使得图形界面的变化只影响数据资源文件的改变，并不需应用程序重新编译链接，极大地提高了开发效率。

3 图形界面程序设计与实现

3.1 标绘台界面主程序设计

我们以 PC/104 模块为嵌入式系统的控制核心，实现所有信息处理和显示功能任务。根据工程应用对标绘台系统的具体要求，应用软件可以划分为以下几个功能模块：串口模块、网络模块、人机交互模块、计算模块、显示模块、看门狗定时器模块，而每一个功能模块又由一个或多个任务组成。其中界面模块是众多模块中最为复杂的模块，包括态势显示、目标绘算、航路导航等界面，在每个单独界面下还包含功能子界面。

利用 Ticon 图形开发工具，按照以下步骤建立图形界面：设置窗体大小、背景等属性；按照标绘系统要求，调整界面整体布局；完善各个功能模块界面属性；运行界面，查看仿真效果。

把设计好的图形界面保存为 twd 文件。点击开发环境自带的测试功能以检验整个图形界面的运行效果，测试结果可以看出该界面是否满足系统要求。

基于 Tilcon 的海图标绘台图形界面应用程序执行流程如图 2 所示，所应用程序设计过程需按照执行流程进行。航海标绘台系统的图形界面应用程序主体由四部分构成：分别是初始化 Tilcon 图形引擎；加载以 twd 文件保存的图形界面；事件处理主循环；任务的发起以及界面程序的退出。下面分别针对上述结构给出设计思路和主要程序段。

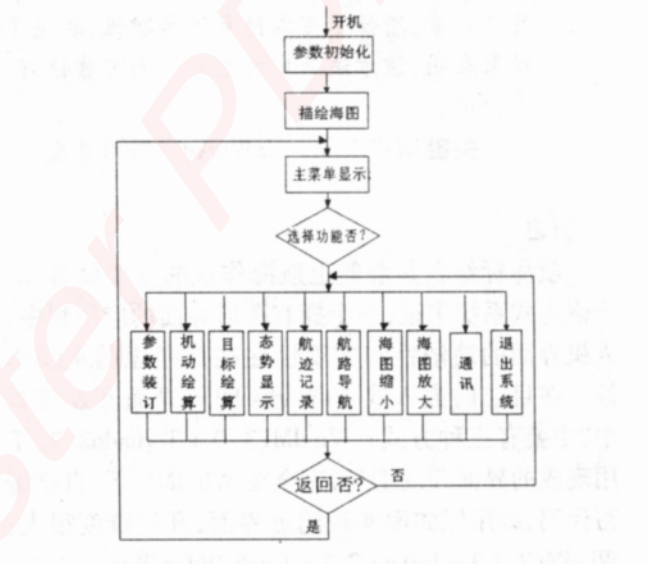

图 2 标绘台流程图

1 初始化 Tilcon 图形引擎

Tilcon 图形引擎初始化的工作是设定操作系统类型，对象空间名和 Licence 文件路径等参数，是整个图形界面运行的前期必备工作。使用 TRT_Strat 系统函数启动一个 Tilcon 图形引擎并与此引擎进行通信，与用户进行交互；

2 加载 twd 图形界面文件

首先指定 twd 窗口文件所在的路径，通过 TRT_WindowLoad 把图形界面资源文件加载到引擎中，然后通过 TRT_windowDisplay 显示到屏幕上，现在便可以 Tilcon 提供的 API 函数进行通信来控制图形界面的显示和刷新功能；

3 事件处理主循环

在主循环当中，利用 TRT_GetInput 函数等待，直到图形引擎通知事件产生，图形引擎发出事件通知时便调用用户自定义的窗口事件处理程序处理事件发出的响应。用户可以利用 TRT_GetInput 函数接收来自其它进程的消息，也可以设置为等待某种事件信号量，使用户程序可以和其它进程之间进行

通信 ；

4 窗口的退出

当不再需要某个正在运行窗口的显示时，可以调用函数 TRT_WindowHide 把窗口隐藏起来，或者调用 TRTWindowDelete 直接把窗口删除。

通过上面内容的介绍，以下给出 Tilcon 界面主程序的基本结构 ：

```
TRT_ReceiveData rec_data ;/* 主程序入口* /
int example (int argc, char * argv [ ])
{startData. Flags = FALSE ;/* 启动 Tilcon 图形* /
errorcode = TRT_Start (&StartData ) ;
TRT_cid = StartData. TRT_CID ;
if (errorcode )
{exit 0( ) ;/*结束退出* / }
/* 加载 Tilcon 资源文件* /
errorcode = TRT_WindowLoad T(RT_cid,
MAIN_WINDOW_FILE ) ;
if (errorcode )
{TRT_Exit T(RT_cid ) ;
exit 0( ) ; }
errorcode = TRT_WindowDisplay T(RT_cid,
MAIN_WINDOW_ID ) ;
if (errorcode )
{TRT_Exit T(RT_cid );/* 出错退出* /
exit 0( ) ; }
/* 主消息循环* /
while (ContinueLooping )
{/* 主消息获取函数* /
c = TRT_GetInput (NULL, 0, NULL, 0,&rec_data,
TRT_BLOCK ) ;
if (c( = =0 )
{/* 主消息处理函数* /
if (rec_data. state = = TRT_window_quit )
{ContinueLooping = FALSE ;
break ; }
}
/* 删除窗口* /
TRT_WindowDelete T(RT_cid,
MAIN_WINDOW_ID ) ;
TRT_Exit T(RT_cid );/*;结束退出* /
return 0( ) ;
}
```

3.2 关于 Tilcon 与 APP 之间的架构方式

在 Tilcon 和 APP 之间的架构方式中，大多数用户都去选择以下两种方式 ：是只利用 TRT_timer-Hint T(ilcon 定时器)定时查询一个数据结构。该数据结构是用户自己在 APP 中针对界面上所有 GUI 控件及其属性建立了一个很庞大的数据结构，根据数据结构的变动来处理 GUI 变化 其二是采用多信道方式，在自己每一个 APP 线程中都启动 Tilcon 引擎，在每个 APP 线程中都直接对 GUI 进行控制和处理。通过应用比较，定时器方式实时性太差只能用于一般对显示要求不太高的系统 而多信道方式，结构实时较强但程序结构混乱，APP 和 GUI 完全绑定，在代码更改某处时很容易牵连到别处。而我们采用双信道通信架构方式，让 GUI 和 APP 建立通信。GUI - > APP 走的是 Tilcon 默认主信道 信道号为 2)也就是 Notification 方式。APP - > GUI 走的是 Tilcon - chanel 方式。该信道发送端位于 APP 线程，接受端位于 Tilcon 线程，其中在 APP 线程中还专门创立了一个子线程 循环阻塞接受一个消息队列 实现消息队列和 Tilcon 自身队列双缓冲的方式 缓冲 APP 线程传递过来的信息。通过 APP - > GUI 传递过来的信息是自定义的协议信息，Tilcon 线程端通过对自定义协议的解析去处理相应的 GUI 控件。在我们项目工程应用中，这种方式得到了可行性验证。

3.3 Tilcon 汉字显示的实现

Tilcon 图形编辑器是支持汉字显示的，但在 VxWorks 系统里界面应用程序 DownLoad 到目标机下运行时，Tilcon 界面的汉字却不能直接显示出来。因为 Tilcon 只能识别 UTF - 8 编码，而在应用程序中汉字是以机器内码表示的。Tilcon 从设计上已经充分考虑到多语言支持的需要，因此其在引擎内部，包括其界面设计工具生成的数据文件中的字符，都以 UNICODE 编码的形式存储。所以我们必须对汉字编码经过两次转换，形成 UTF - 8 编码，Tilcon 才能正确显示汉字。

首先把机器内码转换为 UNICODE 码。UNICODE 目前普遍采用的是 UCS - 2，它用两个字节来编码一个汉字。用户可以设计一个二维表，分别存储汉字机器内码和汉字 UCS - 2 码，机器内码和 UCS - 2 码都可以通过查表得到。构建这样一个二维表格主要是建立汉字内码和 UCS - 2 码之间的对应关系，通过汉字机器内码可以查找到对应的汉字 UCS - 2 码 其次把汉字 UCS - 2 码转换为 UTF - 8 码。在 VxWorks 平台下，由于操作系统本身没有提供 UCS - 2 与本地多字节编码的相互转换函数，因

此在 VxWorks 平台下实现汉字显示不能采用和 Windows 平台那样 Tilcon 通过 Windows 的固有 API 实现 TRT_MBToUTF8,TRT_UTF8ToMB 两个转换函数用以实现编码的相互转换。在 VxWorks 平台下 Tilcon 本身是构架在 WindML 的基础上的,需要 WindML 的支持 另外我们还需要自己编写从 UCS -2 到 UTF - 8 编码的转化实现,下面给出从 UCS - 2 编码到 UTF - 8 的程序代码 :

```
void UTF8FromUCS2 (const wchar_t * uptr, uint tlen, uchar * putf, uint len )
//被汉字转换函数调用的函数
  {
for  i(= 0  ; < tlen && uptr [i]  ;+ +  )
 uch = uptr [i]  ;
if  uch < 0x80  )
 putf [k + +] = uchar )uch( )  ;  }
else if  uch < 0x800  )
 putf [k + +] = uchar )0x0(0 | uch > > 6  )  )  ;
putf [k + +] = uchar )0x8(0 | uch & 0x3f  )  )  ;
else
 putf [k + +] = uchar )0xE(0 | uch > > 12  )  )  ;
putf [k + +] = uchar )0x8(0 | (uch( > > 6 )&
0x3f  )  )  ;
putf [k + +] = uchar )0x8(0 | uch & 0x3f  )  )  ;
  }
putf [k] = ´\0´ ;
  }
void test_Chinesecharacters  (//汉字转换函数
  {
wbuf [0]  = 0x4EBA  /;人 0xE4,0xBA,0xBA
wbuf [i]  = 0x8BA2  /;订 0xE8,0xAE,0xA2
wbuf [2]  = 0x63D0  /;提 0xE6,0x8F,0x90
UTF8FromUCS2  (wbuf, wcslen (wbuf  ),utf8buf, 1024  )  ;
  }
```

4 结束语

Tilcon 作为 WindRiver 新支持的图形界面开发工具,为用户提供了功能齐全的控件,为开发人员提供了便利,将开发人员从繁琐的嵌入式系统界面开发中解脱出来,更好地把精力投入到系统应用功能实现上。在航海标绘台系统中,以 Tilcon 设计用户操控界面,既缩短了软件设计周期,又提高了系统的可靠性和可维护性,具有很高的实用价值。经测试表明,基于 Tilcon 的航海标绘台用户操控界面是比较理想的图形界面应用技术解决方案。

参考文献

[1] 陈智育,温彦军,陈琪. VxWorks 程序开发实践 [M]. 北京 人民邮电出版社, 2004,2,05

[2] 焦永和,冯欣欣. 基于 VxWorks 的中文图形界面开发 [J]. 北京理工大学学报, 2006 ,2 26)

[3] 孔祥营,柏桂枝. 嵌入式实时操作系统 VxWorks 及其开发环境 Tornado [M]. 北京 中国电力出版社, 20021

[4] 王学龙. 嵌入式 VxWorks 系统开发与应用 [M]. 北京 人民邮电出版社, 20031

[5] 王健,赵彬,张宁. 基于 VxWorks 的雷达中央控制系统的设计与实现 [J]. 计算机测量与控制,2003 ,11 6( )

[6] Johnson V. Introducing to IP Muhicast Routing [Z]. Stardust Technology Ine,1997

[7] Nkin. A IETF Criteria for Evaluating Reliable Mul - ticast transport and Application tocols [S],FC2357,998

[8] WindML 3. 0. 1 p rogrammer's guide [Z]. Wind River Systems Inc. 2002

[9] VxWorks programmer's guide 5. 3. 1 [Z]. Wind River Systems Inc. 1997

嵌入式资源免费下载

总线协议：

1. 基于 PCIe 驱动程序的数据传输卡 DMA 传输
2. 基于 PCIe 总线协议的设备驱动开发
3. CANopen 协议介绍
4. 基于 PXI 总线 RS422 数据通信卡 WDM 驱动程序设计
5. FPGA 实现 PCIe 总线 DMA 设计
6. PCI Express 协议实现与验证
7. VPX 总线技术及其实现
8. 基于 Xilinx FPGA 的 PCIE 接口实现
9. 基于 PCI 总线的 GPS 授时卡设计
10. 基于 CPCI 标准的 6U 信号处理平台的设计
11. USB30 电路保护
12. USB30 协议分析与框架设计
13. USB 30 中的 CRC 校验原理及实现
14. 基于 CPLD 的 UART 设计
15. IPMI 在 VPX 系统中的应用与设计
16. 基于 CPCI 总线的 PMC 载板设计
17. 基于 VPX 总线的工件台运动控制系统研究与开发
18. PCI Express 流控机制的研究与实现
19. UART16C554 的设计
20. 基于 VPX 的高性能计算机设计
21. 基于 CAN 总线技术的嵌入式网关设计
22. Visual C 串行通讯控件使用方法与技巧的研究
23. IEEE1588 精密时钟同步关键技术研究
24. GPS 信号发生器射频模块的一种实现方案
25. 基于 CPCI 接口的视频采集卡的设计
26. 基于 VPX 的 3U 信号处理平台的设计
27. 基于 PCI Express 总线 1394b 网络传输系统 WDM 驱动设计
28. AT89C52 单片机与 ARINC429 航空总线接口设计
29. 基于 CPCI 总线多 DSP 系统的高速主机接口设计
30. 总线协议中的 CRC 及其在 SATA 通信技术中的应用
31. 基于 FPGA 的 SATA 硬盘加解密控制器设计
32. Modbus 协议在串口通讯中的研究及应用
33. 高可用的磁盘阵列 Cache 的设计和实现
34. RAID 阵列中高速 Cache 管理的优化

35. 一种新的基于 RAID 的 CACHE 技术研究与实现
36. 基于 PCIE-104 总线的高速数据接口设计
37. 基于 VPX 标准的 RapidIO 交换和 Flash 存储模块设计
38. 北斗卫星系统在海洋工程中的应用
39. 北斗卫星系统在远洋船舶上应用的研究
40. 基于 CPCI 总线的红外实时信号处理系统
41. 硬件实现 RAID 与软件实现 RAID 的比较
42. 基于 PCI Express 总线系统的热插拔设计
43. 基于 RAID5 的磁盘阵列 Cache 的研究与实现
44. 基于 PCI 总线的 MPEG2 码流播放卡驱动程序开发
45. 基于磁盘异或引擎的 RAID5 小写性能优化
46. 基于 IEEE1588 的时钟同步技术研究
47. 基于 Davinci 平台的 SD 卡读写优化
48. 基于 PCI 总线的图像处理及传输系统的设计
49. 串口和以太网通信技术在油液在线监测系统中的应用
50. USB30 数据传输协议分析及实现
51. IEEE 1588 协议在工业以太网中的实现
52. 基于 USB30 的设备自定义请求实现方法
53. IEEE1588 协议在网络测控系统中的应用
54. USB30 物理层中弹性缓冲的设计与实现
55. USB30 的高速信息传输瓶颈研究
56. 基于 IPv6 的 UDP 通信的实现
57. 一种基于 IPv6 的流媒体传送方案研究与实现
58. 基于 IPv4-IPv6 双栈的 MODBUS-TCP 协议实现
59. RS485CAN 网关设计与实现
60. MVB 周期信息的实时调度
61. RS485 和 PROFINET 网关设计
62. 基于 IPv6 的 Socket 通信的实现
63. MVB 网络重复器的设计
64. 一种新型 MVB 通信板的探究
65. 具有 MVB 接口的输入输出设备的分析
66. 基于 STM32 的 GSM 模块综合应用
67. 基于 ARM7 的 MVB CAN 网关设计
68. 机车车辆的 MVB CAN 总线网关设计
69. 智能变电站冗余网络中 IEEE1588 协议的应用

VxWorks：

1. 基于 VxWorks 的多任务程序设计

2. 基于 VxWorks 的数据采集存储装置设计
3. Flash 文件系统分析及其在 VxWorks 中的实现
4. VxWorks 多任务编程中的异常研究
5. VxWorks 应用技巧两例
6. 一种基于 VxWorks 的飞行仿真实时管理系统
7. 在 VxWorks 系统中使用 TrueType 字库
8. 基于 FreeType 的 VxWorks 中文显示方案
9. 基于 Tilcon 的 VxWorks 简单动画开发
10. 基于 Tilcon 的某武器显控系统界面设计
11. 基于 Tilcon 的综合导航信息处理装置界面设计
12. VxWorks 的内存配置和管理
13. 基于 VxWorks 系统的 PCI 配置与应用
14. 基于 MPC8270 的 VxWorks BSP 的移植
15. Bootrom 功能改进经验谈
16. 基于 VxWorks 嵌入式系统的中文平台研究与实现
17. VxBus 的 A429 接口驱动
18. 基于 VxBus 和 MPC8569E 千兆网驱动开发和实现
19. 一种基于 vxBus 的 PPC 与 FPGA 高速互联的驱动设计方法
20. 基于 VxBus 的设备驱动开发
21. 基于 VxBus 的驱动程序架构分析
22. 基于 VxBus 的高速数据采集卡驱动程序开发
23. Vxworks 下的冗余 CAN 通讯模块设计
24. WindML 工业平台下开发 S1d13506 驱动及显示功能的实现
25. WindML 中 Mesa 的应用
26. VxWorks 下图形用户界面开发中双缓冲技术应用
27. VxWorks 上的一种 GUI 系统的设计与实现
28. VxWorks 环境下 socket 的实现
29. VxWorks 的 WindML 图形界面程序的框架分析
30. VxWorks 实时操作系统及其在 PC104 下以太网编程的应用
31. 实时操作系统任务调度策略的研究与设计
32. 军事指挥系统中 VxWorks 下汉字显示技术
33. 基于 VxWorks 实时控制系统中文交互界面开发平台
34. 基于 VxWorks 操作系统的 WindML 图形操控界面实现方法
35. 基于 GPU FPGA 芯片原型的 VxWorks 下驱动软件开发
36. VxWorks 下的多串口卡设计
37. VxWorks 内存管理机制的研究
38. T9 输入法在 Tilcon 下的实现
39. 基于 VxWorks 的 WindML 图形界面开发方法
40. 基于 Tilcon 的 IO 控制板可视化测试软件的设计和实现
41. 基于 VxWorks 的通信服务器实时多任务软件设计
42. 基于 VXWORKS 的 RS485MVB 网关的设计与实现
43. 实时操作系统 VxWorks 在微机保护中的应用

44. 基于 VxWorks 的多任务程序设计及通信管理
45. 基于 Tilcon 的 VxWorks 图形界面开发技术
46. 嵌入式图形系统 Tilcon 及应用研究
47. 基于 VxWorks 的数据采集与重演软件的图形界面的设计与实现
48. 基于嵌入式的 Tilcon 用户图形界面设计与开发
49. 基于 Tilcon 的交互式多页面的设计
50. 基于 Tilcon 的嵌入式系统人机界面开发技术
51. 基于 Tilcon 的指控系统多任务人机交互软件设计

Linux：

1. Linux 程序设计第三版及源代码
2. NAND FLASH 文件系统的设计与实现
3. 多通道串行通信设备的 Linux 驱动程序实现
4. Zsh 开发指南-数组
5. 常用 GDB 命令中文速览
6. 嵌入式 C 进阶之道
7. Linux 串口编程实例
8. 基于 Yocto Project 的嵌入式应用设计
9. Android 应用的反编译
10. 基于 Android 行为的加密应用系统研究
11. 嵌入式 Linux 系统移植步步通
12. 嵌入式 CC++语言精华文章集锦
13. 基于 Linux 的高性能服务器端的设计与研究
14. S3C6410 移植 Android 内核
15. Android 开发指南中文版
16. 图解 Linux 操作系统架构设计与实现原理（第二版）
17. 如何在 Ubuntu 和 Linux Mint 下轻松升级 Linux 内核
18. Android 简单 mp3 播放器源码
19. 嵌入式 Linux 系统实时性的研究
20. Android 嵌入式系统架构及内核浅析
21. 基于嵌入式 Linux 操作系统内核实时性的改进方法研究
22. Linux TCP IP 协议详解
23. Linux 桌面环境下内存去重技术的研究与实现
24. 掌握 Android 7.0 新增特性 Quick Settings
25. Android 应用逆向分析方法研究
26. Android 操作系统的课程教学
27. Android 智能手机操作系统的研究
28. Android 英文朗读功能的实现

29. 基于 Yocto 订制嵌入式 Linux 发行版
30. 基于嵌入式 Linux 的网络设备驱动设计与实现
31. 如何高效学习嵌入式
32. 基于 Android 平台的 GPS 定位系统的设计与实现
33. LINUX ARM 下的 USB 驱动开发
34. Linux 下基于 I2C 协议的 RTC 驱动开发
35. 嵌入式下 Linux 系统设备驱动程序的开发
36. 基于嵌入式 Linux 的 SD 卡驱动程序的设计与实现
37. Linux 系统中进程调度策略
38. 嵌入式 Linux 实时性方法
39. 基于实时 Linux 计算机联锁系统实时性分析与改进
40. 基于嵌入式 Linux 下的 USB30 驱动程序开发方法研究
41. Android 手机应用开发之音乐资源播放器
42. Linux 下以太网的 IPv6 隧道技术的实现
43. Research and design of mobile learning platform based on Android
44. 基于 linux 和 Qt 的串口通信调试器调的设计及应用
45. 在 Linux 平台上基于 QT 的动态图像采集系统的设计
46. 基于 Android 平台的医护查房系统的研究与设计
47. 基于 Android 平台的软件自动化监控工具的设计开发
48. 基于 Android 的视频软硬解码及渲染的对比研究与实现
49. 基于 Android 移动设备的加速度传感器技术研究
50. 基于 Android 系统振动测试仪研究
51. 基于缓存竞争优化的 Linux 进程调度策略

Windows CE：

1. Windows CE.NET 下 YAFFS 文件系统 NAND Flash 驱动程序设计
2. Windows CE 的 CAN 总线驱动程序设计
3. 基于 Windows CE.NET 的 ADC 驱动程序实现与应用的研究
4. 基于 Windows CE.NET 平台的串行通信实现
5. 基于 Windows CE.NET 下的 GPRS 模块的研究与开发
6. win2k 下 NTFS 分区用 ntldr 加载进 dos 源代码
7. Windows 下的 USB 设备驱动程序开发
8. WinCE 的大容量程控数据传输解决方案设计
9. WinCE6.0 安装开发详解
10. DOS 下仿 Windows 的自带计算器程序 C 源码
11. G726 局域网语音通话程序和源代码
12. WinCE 主板加载第三方驱动程序的方法
13. WinCE 下的注册表编辑程序和源代码

14. WinCE 串口通信源代码
15. WINCE 的 SD 卡程序[可实现读写的源码]
16. 基于 WinCE 的 BootLoader 研究
17. Windows CE 环境下无线网卡的自动安装
18. 基于 Windows CE 的可视电话的研究与实现
19. 基于 WinCE 的嵌入式图像采集系统设计
20. 基于 ARM 与 WinCE 的掌纹鉴别系统
21. DCOM 协议在网络冗余环境下的应用
22. Windows XP Embedded 在变电站通信管理机中的应用
23. XPE 在多功能显控台上的开发与应用
24. 基于 Windows XP Embedded 的 LKJ2000 仿真系统设计与实现
25. 虚拟仪器的 Windows XP Embedded 操作系统开发
26. 基于 EVC 的嵌入式导航电子地图设计
27. 基于 XPEmbedded 的警务区 SMS 指挥平台的设计与实现
28. 基于 XPE 的数字残币兑换工具开发

PowerPC：

1. Freescale MPC8536 开发板原理图
2. 基于 MPC8548E 的固件设计
3. 基于 MPC8548E 的嵌入式数据处理系统设计
4. 基于 PowerPC 嵌入式网络通信平台的实现
5. PowerPC 在车辆显控系统中的应用
6. 基于 PowerPC 的单板计算机的设计
7. 用 PowerPC860 实现 FPGA 配置
8. 基于 MPC8247 嵌入式电力交换系统的设计与实现
9. 基于设备树的 MPC8247 嵌入式 Linux 系统开发
10. 基于 MPC8313E 嵌入式系统 UBoot 的移植
11. 基于 PowerPC 处理器 SMP 系统的 UBoot 移植
12. 基于 PowerPC 双核处理器嵌入式系统 UBoot 移植
13. 基于 PowerPC 的雷达通用处理机设计
14. PowerPC 平台引导加载程序的移植
15. 基于 PowerPC 嵌入式内核的多串口通信扩展设计
16. 基于 PowerPC 的多网口系统抗干扰设计
17. 基于 MPC860T 与 VxWorks 的图形界面设计
18. 基于 MPC8260 处理器的 PPMC 系统
19. 基于 PowerPC 的控制器研究与设计
20. 基于 PowerPC 的模拟量输入接口扩展

21. 基于 PowerPC 的车载通信系统设计
22. 基于 PowerPC 的嵌入式系统中通用 IO 口的扩展方法
23. 基于 PowerPC440GP 型微控制器的嵌入式系统设计与研究
24. 基于双 PowerPC 7447A 处理器的嵌入式系统硬件设计
25. 基于 PowerPC603e 通用处理模块的设计与实现
26. 嵌入式微机 MPC555 驻留片内监控器的开发与实现
27. 基于 PowerPC 和 DSP 的电能质量在线监测装置的研制
28. 基于 PowerPC 架构多核处理器嵌入式系统硬件设计
29. 基于 PowerPC 的多屏系统设计
30. 基于 PowerPC 的嵌入式 SMP 系统设计

ARM：

1. 基于 DiskOnChip 2000 的驱动程序设计及应用
2. 基于 ARM 体系的 PC-104 总线设计
3. 基于 ARM 的嵌入式系统中断处理机制研究
4. 设计 ARM 的中断处理
5. 基于 ARM 的数据采集系统并行总线的驱动设计
6. S3C2410 下的 TFT LCD 驱动源码
7. STM32 SD 卡移植 FATFS 文件系统源码
8. STM32 ADC 多通道源码
9. ARM Linux 在 EP7312 上的移植
10. ARM 经典 300 问
11. 基于 S5PV210 的频谱监测设备嵌入式系统设计与实现
12. Uboot 中 start.S 源码的指令级的详尽解析
13. 基于 ARM9 的嵌入式 Zigbee 网关设计与实现
14. 基于 S3C6410 处理器的嵌入式 Linux 系统移植
15. CortexA8 平台的μC-OSⅡ及 LwIP 协议栈的移植与实现
16. 基于 ARM 的嵌入式 Linux 无线网卡设备驱动设计
17. ARM S3C2440 Linux ADC 驱动
18. ARM S3C2440 Linux 触摸屏驱动
19. Linux 和 Cortex-A8 的视频处理及数字微波传输系统设计
20. Nand Flash 启动模式下的 Uboot 移植
21. 基于 ARM 处理器的 UART 设计
22. ARM CortexM3 处理器故障的分析与处理
23. ARM 微处理器启动和调试浅析
24. 基于 ARM 系统下映像文件的执行与中断运行机制的实现
25. 中断调用方式的 ARM 二次开发接口设计

26. ARM11 嵌入式系统 Linux 下 LCD 的驱动设计
27. Uboot 在 S3C2440 上的移植
28. 基于 ARM11 的嵌入式无线视频终端的设计
29. 基于 S3C6410 的 Uboot 分析与移植
30. 基于 ARM 嵌入式系统的高保真无损音乐播放器设计
31. UBoot 在 Mini6410 上的移植
32. 基于 ARM11 的嵌入式 Linux NAND FLASH 模拟 U 盘挂载分析与实现
33. 基于 ARM11 的电源完整性分析
34. 基于 ARM S3C6410 的 uboot 分析与移植
35. 基于 S5PC100 移动视频监控终端的设计与实现

Hardware：

1. DSP 电源的典型设计
2. 高频脉冲电源设计
3. 电源的综合保护设计
4. 任意波形电源的设计
5. 高速 PCB 信号完整性分析及应用
6. DM642 高速图像采集系统的电磁干扰设计
7. 使用 COMExpress Nano 工控板实现 IP 调度设备
8. 基于 COM Express 架构的数据记录仪的设计与实现
9. 基于 COM Express 的信号系统逻辑运算单元设计
10. 基于 COM Express 的回波预处理模块设计
11. 基于 X86 平台的简单多任务内核的分析与实现
12. 基于 UEFI Shell 的 PreOS Application 的开发与研究
13. 基于 UEFI 固件的恶意代码防范技术研究
14. MIPS 架构计算机平台的支持固件研究
15. 基于 UEFI 固件的攻击验证技术研究
16. 基于 UEFI 的 Application 和 Driver 的分析与开发
17. 基于 UEFI 的可信 BIOS 研究与实现
18. 基于 UEFI 的国产计算机平台 BIOS 研究
19. 基于 UEFI 的安全模块设计分析
20. 基于 FPGA NiosⅡ的等精度频率计设计
21. 基于 FPGA 的 SOPC 设计
22. 基于 SOPC 基本信号产生器的设计与实现
23. 基于龙芯平台的 PMON 研究与开发
24. 基于 X86 平台的嵌入式 BIOS 可配置设计
25. 基于龙芯 2F 架构的 PMON 分析与优化
26. CPU 与 GPU 之间接口电路的设计与实现

27. 基于龙芯 1A 平台的 PMON 源码编译和启动分析
28. 基于 PC104 工控机的嵌入式直流监控装置的设计
29. GPGPU 技术研究与发展
30. GPU 实现的高速 FIR 数字滤波算法
31. 一种基于 CPUGPU 异构计算的混合编程模型
32. 面向 OpenCL 模型的 GPU 性能优化
33. 基于 GPU 的 FDTD 算法
34. 基于 GPU 的瑕疵检测
35. 基于 GPU 通用计算的分析与研究
36. 面向 OpenCL 架构的 GPGPU 量化性能模型
37. 基于 OpenCL 的图像积分图算法优化研究
38. 基于 OpenCL 的均值平移算法在多个众核平台的性能优化研究
39. 基于 OpenCL 的异构系统并行编程
40. 嵌入式系统中热备份双机切换技术研究

Programming:

1. 计算机软件基础数据结构 - 算法
2. 高级数据结构对算法的优化
3. 零基础学算法
4. Linux 环境下基于 TCP 的 Socket 编程浅析
5. Linux 环境下基于 UDP 的 socket 编程浅析
6. 基于 Socket 的网络编程技术及其实现
7. 数据结构考题 - 第 1 章 绪论
8. 数据结构考题 - 第 2 章 线性表
9. 数据结构考题 - 第 2 章 线性表 - 答案

FPGA / CPLD:

1. 一种基于并行处理器的快速车道线检测系统及 FPGA 实现
2.