一种新型 MVB 通信板的探究

王　雪，申　萍，宋　娟，严　翔
（北京交通大学　电气工程学院，北京 100044）

摘　要　结合车载安全计算机技术的发展，以 ARM7 系列为核心控制器，采用 SOPC 技术，设计了一种全新的 VME 总线接口 MVB 通信板卡。重点介绍了基于 USB 通信接口，使用 VC++6.0 开发的上位机试验环境及试验结果。

关键词　MVB；VME；VC++
中图分类号：U264.6　　**文献标志码**：A

随着我国高速铁路的不断发展，列车控制系统的高安全性和高可靠性已成为轨道交通领域中尤为关键的指标。VME(Versa Module Eurocard)总线为一种通用的计算机总线，定义了一个在紧密耦合硬件构架中可进行互联数据处理、数据存储和连接外围控制器件的结构，1987 年被批准为国际标准 IEEE 1014。其机械结构具有良好的抗震性和抗冲击能力，国外，尤其是部分欧洲国家已将其成熟地应用于车载安全计算机内，以保证安全性和可靠性，而国内相关研究起步较晚，在此基础上开发的列车网络设备产品也相对稀少。

提出一种具有 VME 接口的 MVB 通信板卡设计与实现方案，以解决 TCN 与车载安全计算机互联的问题，满足该类技术产品国产化的迫切需要。利用此板卡可以实现网络主从设备的数据传输、节点组网、接入 MVB 网络等功能，也为国内车辆使用 MVB 技术和开发基于 MVB 的其他应用提供平台。

1　系统组成

系统整体构架如图 1 所示，包括 ARM 主控制器、VME 总线接口控制器、MVB 总线控制器、电源复位和 JTAG 电路 5 部分。

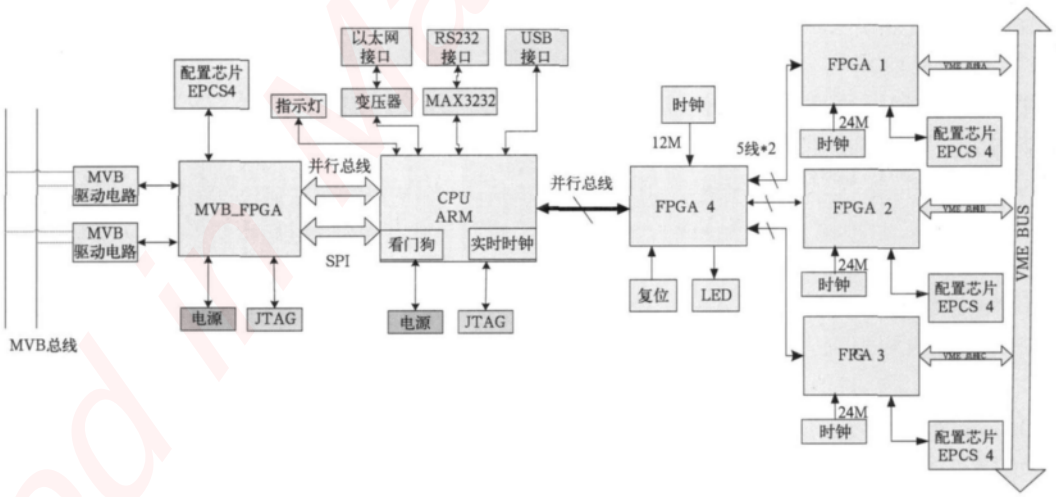

图 1　系统整体设计方案框图

1.1　CPU 选型

为实现 MVB 与 VME 总线的通信，系统选用 ARM 作为高速中央处理器进行控制，具有集成化、小型化、实时性强、可移植性强、低成本和低功耗的优势。主控制器选用 NXP 公司的 LPC2478。此款 ARM 芯片处理器时钟高达 72 MHz，片内 SRAM 资源达 98 kb：4 kb 支持高性能 CPU 通过 ARM 局部总线访问，2 kb 用于 RTC 供电以便存储数据，16 kb 作为通用 SRAM 或用于以太网接口，另 16 kb 可供 GPDMA 使用或 USB 访问。此外，还设计扩展了 64 Mb 的 SDRAM 和 Samsung 公司的 256 Mb 的 NAND Flash。

片上集成了 10/100M 以太网媒体访问控制器

(MAC)、USB2.0 全速 Device/Host/OTG 控制器、4 个 UART 和 1 个 SPI 接口等。CPU 外围还需对这些接口电路进行设计，另外，控制 I/O(并行总线、指示灯等)电路及 JTAG 菊花链下载调试电路的设计也必不可少。

1.2 VME 总线接口设计

VME 总线因其并行性、实时性、高可靠性等特点，在高性能的实时应用领域一直处于主导地位，已经发展到了一个比较完善的阶段。国外许多电动车组将其作为车载安全计算机系统的板级总线，并应用于车载网络设备上。设计并开发 MVB 通信板上的 VME 总线接口，有助于实现与该类引进产品的板级互换。

VME 总线数据传输采用 32 位地址及 32 位数据非复用结构，支持 8 位、16 位和 32 位的数据传输。异步通信，不需要时钟同步，利用 DTACK＊握手信号来控制传输过程。按功能分为 4 类总线：数据传输总线、数据传输仲裁总线、优先权中断总线和公用信号线。为完成基本接口功能，对其信号线进行剪裁，保留 36 根：数据线 D[7:0]，地址线 A[15:1]，数据选通 DS0＊和 DS1＊，地址选通 AS＊，读/写信号 WRITE＊，传输应答 DTACK＊，长字信号 LWORD＊，插槽编码 GA[4:0]和 GAP＊，系统复位 SYRST＊。其中，依靠主模块驱动 DS0＊，DS1＊，LWORD＊和地址线 A01 的不同组合，控制数据的传输周期类型。

另外，为了提高系统的可靠性和安全性，车载安全计算机系统通常采用双机热备、2 取 2 乘 2、3 取 2 等冗余结构。IEC 61508 标准应用马尔可夫模型分别对上述结构的可靠度、安全度和可用度进行了计算比较，性能最高的 2 种结构为 3 取 2 和 2 取 2 乘 2。考虑到系统的复杂度和车内空间限制，本设计采用 3 取 2 构架，通信板上与安全计算机互联一侧设计 3 路 VME 总线并行工作：FPGA1、2、3 分别完成对应的 VME_A、VME_B、VME_C 3 路背板总线接口逻辑，FPGA4 作为 3 取 2 表决器对运算结果进行两两表决，最终将一致的数据存储于双口 RAM 中，以便主控制器 ARM 的访问。当其中一路信号与另外两路不同时，屏蔽该路，剩余两路进行 2 取 2 表决，使得系统在降级工作的情况下也能保证安全输出；只有当 3 路信号互不相同时，系统才判断错误，不进行输出。任何一条电路或单个元件的故障都不会影响整个系统的正常运行。

VME 国际标准规定，6U 板卡可以设置 1 或 2 个 VEM 总线连接器。该通信板卡设计为双连接器，采用欧式规范 DIN 41612，P1/J1 在上(96 针)，P2/J2 在下(扩展为 160 针)，便于连接不同类型背板机箱。

1.3 MVB 总线接口设计

MVB 总线控制器用于实现物理层信号的转换，执行数据链路层的通信规程，由试验室自主研发的 SOPC 技术实现。通过单片 FPGA 将 MVB 主设备或其他网络设备发送的数据传给 MVB 网络，或从 MVB 网络中接收其他设备发出的数据并送给主设备。数据传输主要采用并行总线的接口模式，另外增加了 SPI 串行总线作为备用的数据访问方式。并行总线包括 16 位数据线、16 位地址线、读/写、使能信号等。

2 通信板的试验研究

2.1 MVB 与 VME 间通信

MVB 与 VME 总线上的数据交换主要是依靠主控制器 ARM 访问封装在各自 FPGA 中双口 RAM 完成的。ARM 作为控制核心，决定访问 MVB 总线或是 VME 总线上的数据，并将数据存储在 SDRAM/FLASH 内。

FPGA4 内封装了一个片内 RAM，当 VME 总线发送数据包时，FPGA1～3 对其读取。定义数据高 8 位为标志信息，低 8 位为实际数据，经过接口模块将其按地址映射于 FPGA1～3 片内 RAM 区域中，所得数据再经逻辑变换为串行数据，经过自定义串行总线传输至 FPGA4 的 FIFO 内。FPGA4 对整个数据包 3 取 2 判别后，分别提取出相应的地址信息和数据信息。每当堆栈中存满 16 位时，FPGA4 的 FIFO 向片内 RAM 发送数据包，等待 ARM 来读取。表决芯片 FPGA4 与 FPGA1～3 之间自定义 5 路双向的串行总线：1 路时钟(CLK)，1 路启动信号(START)，3 路收发数据(DATA)。实际只采取了 1 路 DATA，即共有 3 路双向串行总线工作。

ARM 控制器读取 VME 数据时，根据地址映射到 FPGA4 的片内 RAM 空间，通过 8 位并行总线从 FPGA4 片内 RAM 里获得。ARM 控制器经协议转换将信号输出，经过并行总线传输给 MVB_FPGA，以数据地址总线访问的方式交换数据。封装于其中的 MVBC 将信号经过逻辑变换和驱动电路后传递到 MVB 总线上。

设计中，MVB_FPGA 植入试验室自主开发的 MVB IP 核，用单片 FPGA 实现 MVB 的所有功能，代替现有网卡的 MVB 芯片，支持过程数据、消息数据、监控数据报文的形成和处理，还可以实现总线管理性能，进而成为 MVB 设备分类中的 4 类设备。

2.2 通信板与 PC 机间通信

通信板与 PC 机之间的通信，可以分别在 ADS1.2 开发环境下通过 RS232 串口、USB2.0 接口和以太网接口完成。结合 LPC2478 ARM 资源，本文主要针对 USB 接口进行开发。

LPC2478 ARM 内部自带一个 USB 设备控制器，支持 32 个固定配置的物理端点(16 个逻辑端点)，并完全兼容 USB2.0 全速规范，使 USB 设备控制器与 CPU 间的数据交换稳定地达到很高的速度。数据传输模式

采用从机模式(非 DMA 模式),由 USB 设备控制器向主机 ARM MCU 提交中断的机制来完成。USB 固件总是在等待主机的命令,然后根据命令去执行相应的程序。如果 LPC2478 USB 发现从 USB 总线上收到数据,USB 设备控制器发生中断,进入 USB 中断服务程序→中断服务程序读取数据,并置"端点收到数据标志"位为 1,然后退出中断回到前台→前台检测到"端点收到数据标志"位为 1 时,针对接收到的数据进行处理;如果有数据需要发送到 USB 主机,只须将要发送的数据写入 LPC2478 USB 发送端点缓冲区中。ARM MCU 前台不断循环处理相关事情,后台进行 LPC2400 USB 中断服务程序,从而完成与 PC 机的底层通信。

2.3 上位机试验环境开发

对于开发者来讲,通信是否正常需要通过上位机验证;对于用户而言,与系统的交互也是通过上位机实现的。本系统在 VC++6.0 MFC 环境下对上位机应用程序进行了开发,利用 Windows API 实现 USB 接口通信,对接入系统内 MVB 网络上的数据进行监控及配置。具体实现以下功能:

(1) 实现 MVB 端口号、功能码、设备主从,设备状态等配置信息的下载;

(2) 过程数据端口可进行动态配置;

(3) 已配置的端口信息可分别进入自动分配模式和手动分配模式;

(4) 可对已分配的端口进行数据发送和接收;

(5) 可进行消息数据的配置、发送及监视等。

2.3.1 USB 驱动

本设计采用了一个基于 Windows API 函数 I/O 请求的动态链接库 DLL 文件,作为应用层与 USB 驱动程序的接口,在 VC++6.0 开发环境下,通过调用这个 DLL 文件实现与 USB 互通。利用这种方式,使进程可以调用本不属于其可执行代码的函数,有助于节省内存和提高数据和资源的共享度,简化了上位机软件的管理,同时也减少了开发难度,缩短了开发周期。

该动态链接库为用户开放了以下几种的 API 函数:

Int __stdcall LPC2400_ReadData(int siPipeNum, unsigned char * pucRcvBuf, int siReadLen, int siWaitTime); //从 USB 设备读取数据(管道号为偶)

Int __stdcall LPC2400_WriteData(int siPipeNum, unsigned char * pucSendBuf, int siSendLen, int siWaitTime); //向 USB 设备写入数据(管道号为奇)

Int __stdcall LPC2400_ReadPortX(unsigned char * pucRcvBuf, int siReadLen, int waittime=-1); //从 USB 设备的逻辑端点 X 读数据

Int __stdcall LPC2400_WritePortX(unsigned char * pucSendBuf, int siSendLen, int siWaitTime=-1); //向 USB 设备的逻辑端点 X 写数据

2.3.2 MVB 配置

对 MVB 通信板卡的配置都是由上位机软件实现的。根据 MVB 通信协议,软件初始界面设计分为两个部分,设备配置和端口过程数据配置,如图 2 所示。

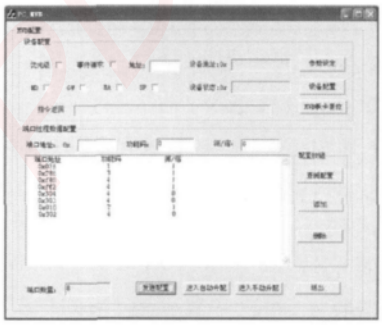

图 2 MVB 端口信息配置界面

(1) 设备配置

在【设备地址】和【设备状态】可分别显示组合后 2 字节的 16 位数据设备信息。单击"设备配置"对设备进行配置,发送帧如下:"0xaa, 0x55, 0x11, Len(数据内容长度),设备地址(2 Bytes),设备状态(2 Bytes)"。

配置成功后设备区域将变为不可编辑状态,并将所发数据包通过 ARM 返回给上位机。若配置失败可弹出错误提示。

(2) 端口过程数据配置

在端口配置信息列表中,连接了 ACCESS 数据库,列表可按"端口地址"、"功能码"和"源/宿端口属性"的顺序逐条显示保存在数据库中的信息。ACCESS 数据库的连接和打开,需要注意在头文件中定义相关结构体,然后在初始化 MVB 配置对话框时实现。

过程数据端口配置完毕后,点击"发送配置",则发送数据包格式如下:

"0xaa, 0x55, 0x22, Len,端口 0 信息,端口 1 信息……. 端口 N-1 信息"

每个端口信息组合如表 1。

表 1 端口信息组合

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0×00	Addr_7	Addr_6	Addr_5	Addr_4	Addr_3	Addr_2	Addr_1	Addr_0
0×01	SRC/ SINK	Fcode[2]	Fcode [1]	Fcode [0]	Addr_11	Addr_10	Addr_9	Addr_8

Addr_0～Addr_11 是端口地址；Fcode[0]～Fcode[2] 是功能码设定，由于此处配置都为过程数据，所以仅 0～4 有效；SRC/ SINK 是端口源/宿标志，1 表示是源端口，0 表示是宿端口。

2.3.3 数据收发

端口配置完毕后，就可以选择“进入自动分配”或“进入手动分配”分别进入端口自动分配模式和手动选择模式。新对话框的弹出需要对其进行定义。

另外，为了方便相关参数和函数在各个对话框中可以共享，引入了全局变量的使用。全局变量的定义必须放在恰当的文件中。在此实例中，这些全局变量的定义应放在主对话框的实现文件 PC_MVBDlg. cpp 中，具体的位置应在包含文件和条件编译之后，且在所有成员函数之外，这样才可以在此文件的成员函数中使用它们。同时，为了使其中一些全局变量可以被子对话框使用，应在相应的子对话框实现文件（如 dataDlg. cpp）中进行 extern 变量声明。

如图 3 所示，在自动分配模式界面中，仅支持前 10 个端口的数据收发，将已配置的端口地址逐条显示出来。功能码不同，可以发送和接收的数据长度也不同。若为源端口，按键显示“发送”，根据提示的数据长度填入数据后（长度不足的自动添零），点击此按钮，该端口的数据将会发送到 MVB 网络上；若为宿端口，按键显示“接收”，直接点击此按钮，将会显示接收到的该端口网络数据。

手动分配模式支持全部已配置端口的数据收发，并含有消息数据配置、发送及监视等功能。如图 4 所示，进入手动分配界面后，系统会自动将已配置的所有端口按源/宿进行分类。在过程数据部分，输入任意已有的源端口地址或宿端口地址，即可进行数据的收发。

图 3 过程数据端口自动分配界面

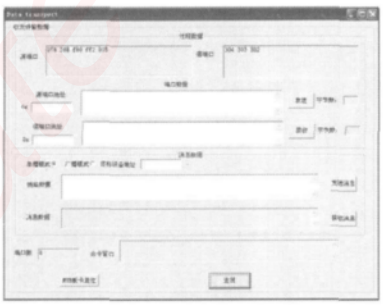

图 4 过程数据端口手动分配界面

不论是自动分配模式还是手动分配模式，如果在设备运行过程中需要重新对 MVB 过程数据端口进行配置，可使用“MVB 板卡复位”按键，这样重配置后的参数才会生效。

3 试验测试结果

为测试 MVB 网络及上位机的通信情况，进行自组网试验进行验证。将设备 A、设备 B 和 MVB 主设备联网，如图 5 所示。分别通过上位机对设备 A 和设备 B 进行配置。在端口配置中，两个设备要分别含有对应的源端口和宿端口。

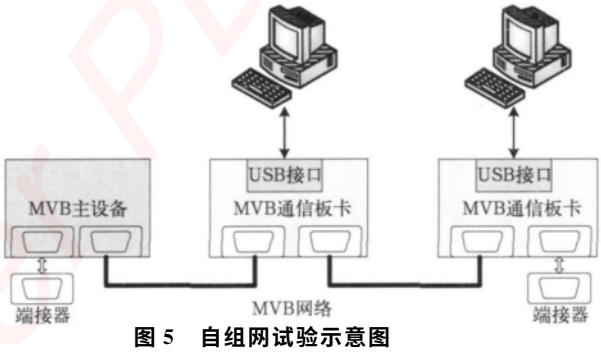

图 5 自组网试验示意图

如图 6、图 7 所示，设备 A 用源端口 0×10 发送了数据，设备 B 在宿端口 0×10 接收到了数据，数据一致；设备 B 用源端口 0×302 发送了数据，设备 A 在宿端口 0×302，接收到了数据，数据也一致，说明过程数据通信正常。类似的，消息数据通信也可以由此验证。

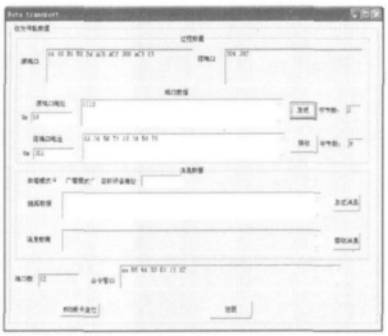

图 6 设备 A 收发过程数据

图 7 设备 B 收发过程数据

HX_D3C 型电力机车辅助变流器水冷却技术的研究与实现

李　新[1]，谢陈刚[2]

（1　中国北车集团　大连机车车辆有限公司，辽宁大连 116022；

2　武汉铁路局　机务处，湖北武汉 430071）

摘　要　HX_D3C 型机车投入运用以来，经常因辅助变流器风冷系统冷却不良造成机破。为了彻底解决该辅助变流器冷却不良的问题，提出了采用主、辅变流器水冷却系统并联、降低辅助变流器开关损耗的总体方案。通过对该方案中功率元件散热、冷却能力等技术参数的分析及试验验证，说明了方案的可行性。该方案能彻底解决风冷散热器污损而引起的机破问题，为后期 HX_D3C 机车辅助变流器水冷方案的批量装车奠定了基础。

关键词　HX_D3C 型电力机车；辅助变流器；水冷却技术

中图分类号：U264.5+5　　**文献标志码**：A

HX_D3C 型机车是在 6 轴大功率交流传动电力机车技术平台的基础上衍生出的具有列车供电功能的交流传动客货通用型电力机车。目前主要在北京、上海、武汉、西安、南昌、济南铁路局、广铁集团公司等担当客运牵引任务。HX_D3C 型机车继承了 HX_D3 型机车的技术体系，具有牵引力大、黏着性能好、恒功率速度范围宽、效率高等特点。该车辅助变流器的初期设计方案，沿用了风冷却方案，在运用中发现，受到机车运用区段环境的影响，风冷系统进风口处的滤网和辅助变流器的散热器经常被灰尘和杂物堵塞，导致冷却能力下降，影响机车运行，甚至造成机破。

李新（1980－）男，辽宁阜新人，工程师（修回日期：2012－04－13）

5　结束语

给出了基于 ARM，且具有 VME 总线接口的 MVB 通信板卡的设计思路与实现方案。重点研究了试验环境的开发过程，以便对 MVB 通信板卡在安全冗余等方面开展研究。试验结果表明，所开发的上位机环境能够很好地对底层 MVB 网络进行配置及数据收发，并监控车载安全计算机通过 VME 总线与 MVB 网络间的通信情况。

参考文献

[1]　IEC 61375-1, Electric railway equipment Train Bus Part1: Train Communication Network[S].

[2]　陈　彦. 基于 VME 总线的 MVB 通信板的设计与开发[D]. 北京：北京交通大学. 2011.

[3]　穆云丽. 多功能接口 MVB 网卡在机车状态监测与故障诊断系统的应用[D]. 北京：北京交通大学. 2009.

[4]　孙　鑫，余安萍. VC++深入详解[M]. 北京：电子工业出版社，2006.

[5]　杨永安，杨　松，孙　丽. 基于 ARM 构架的嵌入式 USB 驱动的设计[J]. 内蒙古农业大学学报. 2008，29(4)：200-202.

[6]　宋红霞，等. 列车自动防护系统安全计算机可靠性与安全性分析[J]. 工业控制计算机. 2008，21(1)：13-15.

Research on a New Type MVB Communication Board

WANG Xue, *SHEN Ping*, *SONG Juan*, *YAN Xiang*

(College of Electrical Engineering, Beijing Jiaotong University, Beijing 100044)

Abstract: According to the development of vehicle safety computer technology, ARM7 as the core controller, with SOPC technology, this paper designs a new kind of MVB communication board with VME bus interface. Especially, it provides the development of PC based on the USB communication interface by VC++ 6.0 and the experimental results.

Key words: MVB; VME; VC++

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

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

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

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

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

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

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

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