

分布式控制系统精确时钟同步技术

Precise Clock Synchronization for Distributed Control System

杨传顺 袁建 李国华

江苏自动化研究所 江苏 连云港 222006)

摘要 为了克服传统的分布式测量和控制系统存在的诸多缺点,提出了传输和同步系统时钟的新方案。该方案采用 IEEE 1588 精确时钟同步协议以及在控制系统中广泛应用的以太网来实现。实现过程中,由硬件辅助生成时间戳,保证了系统对主从时钟的偏移时间和报文传输的延迟时间的测量、计算和精确控制。测试结果表明,从时钟的同步精度能够满足系统的应用需求,减少了测控系统中的专用时钟线缆的连接数,对类似的时钟同步需求具有很好的推广应用价值。

关键词 分布式控制系统 IEEE 1588 精确时钟同步 延迟测量 时间戳

中图分类号 TP273 文献标志码 A

Abstract To overcome various demerits in traditional distributed measurement and control systems, by adopting IEEE 1588 precise clock synchronization protocol the scheme of transmitting and synchronizing the system clock via Ethernet in measurement and control systems is proposed. During implementation, the timestamp is generated in aid of hardware, to guarantee the system precisely measure, calculate, and control the time offset between master and slave clocks, as well as the time delay of message transmission. The test result shows that the synchronous accuracy of the slave clock can satisfy the application requirement of system, and the quantity of the linkages of dedicated wires or cables of clock in measurement and control system is reduced. This technology possesses good popularization and application value for similar clock synchronization requirement.

Keywords Distributed control system (DCS) IEEE 1588 Precise clock synchronization Delay measurement Timestamp

0 引言

随着分布式控制系统应用范围的广泛增长和应用规模的不断扩大,通过系统网络特别是控制系统中广泛应用的以太网传递和同步系统内各分散节点的时钟变得越来越重要。

系统对时钟的应用主要有两种:基于绝对时间的应用和基于频率的应用。绝对时间用于记录系统内事项发生的次序,频率用于控制器、执行器以及传感器的同步采样、反馈和执行。

基于此,工业界一直在致力于解决以太网定时同步能力不足的问题。1992年推出的基于软件控制方式的网络时间协议(network time protocol, NTP)同步精度可以达到200 μ s,但是不能满足测量仪器和工业控制所需的精度。为了提高分布网络中定时同步的精度,美国工业界成立了网络精确时钟同步委员会,于2002年底推出了IEEE 1588标准^[1]。该标准的精确时钟同步协议在时钟同步方面取得了重大进展,2008年

推出了该标准的2.0版,国际电工协会也把此标准纳入到了IEC标准中,对应的标准是IEC 61588^[2]。

1 IEEE 1588 标准

IEEE 1588标准的全称是网络测量和控制系统的精确时钟同步协议(precision time protocol, PTP)。PTP和NTP技术一样,具有配置容易、收敛快速以及对网络带宽和资源消耗少等优点。它的主要原理是通过一个同步信号周期性地对网络中所有节点的时钟进行校正,可以使基于以太网的分布式系统时钟达到高精度同步。

在IEEE 1588标准中,系统内的时钟分为普通时钟和边界时钟两种。只有1个PTP通信端口的时钟是普通时钟,有2个及以上PTP通信端口的时钟是边界时钟。每个PTP端口的状态有9种,主要的3种状态是主状态(PTP_MASTER)、从状态(PTP_SLAVE)和被动状态(PTP_PASSIVE)。PTP端口处于主状态的时钟称为主时钟(grandmaster clock, GMC),处于从状态的时钟称为从时钟,主时钟负责同步系统内所有的从时钟。如果PTP端口处于被动状态,则意味着对应的时钟不参与时钟同步。

理论上任何时钟都能实现主时钟和从时钟的功

能,但一个系统内只能有一个主时钟,整个系统中的最优时钟为主时钟,它具有最好的稳定性和精确性等。根据各节点的时钟精度、级别以及通用协调时间 universal time coordinated, UTC 的可追溯性等,系统采用最优时钟算法来自动选择系统内的主时钟。典型的主从时钟关系示意图如图 1 所示。



图 1 主从时钟关系示意图

Fig. 1 Relationship between master and slave clocks

PTP 的参考体系结构如图 2 所示^[3],协议层包含 3 个通信接口:端口接口、时间戳接口和时钟接口。端口接口用来分配和接收 PTP 信息。时间戳接口通过对同步报文加盖时间戳来实现精确时钟同步,可以根据精度要求决定采用是硬件还是软件产生的时间戳。时钟接口能够对本地图钟进行读写操作,达到绝对时间和频率与主时钟的同步。该接口包含了与时钟同步质量如精确度、稳定度等密切相关的控制算法。

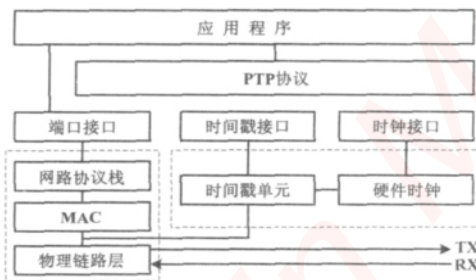


图 2 PTP 参考体系结构

Fig. 2 PTP reference architecture

2 精确时钟同步协议的算法

精确时钟同步协议 PTP 采用分层的主从模式进行时钟同步,同步的基本原理是对发出和接收的报文时间进行记录,每条报文都有时间戳信息。主时钟周期性发布时间同步报文,从时钟根据接收到的同步报文的时间戳信息,计算出主从线路报文传输的时间延迟和主从时钟的时钟偏移,并相应地调整本地时间,使从时钟保持与主时钟一致的频率与相位。系统据此实现主从时钟的频率同步和时间同步。时钟同步的精度主要依赖于计数器频率的准确性和链路的对称性^[4]。

PTP 定义了 4 种多点传送的报文:同步报文、跟随报文、延迟请求报文和延迟应答报文。这些报文的交互顺序如图 3 所示,收到的信息回应与时钟当前的状态有关。

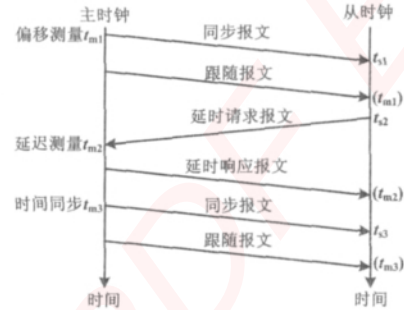


图 3 偏移和延迟的测量

Fig. 3 Measurement of time offset and delay

2.1 偏移和延迟测量

为了描述分布式控制系统的时钟同步问题,我们以一个主时钟和一个从时钟的同步过程为例,其具体算法说明如下。

① 主时钟按照定义的间隔时间(缺省是 2 s)周期性地向从时钟发送“同步报文”,这个报文的时间戳是主时钟打上的预计发送时间。该报文在接收端由从时钟记录下接收时间 t_{s1} 。PTP 支持由硬件辅助在物理层或接近物理层的位置检测和记录时间戳,以便提高同步时钟的精度。

② 主时钟向从时钟发送“跟随报文”,此报文包含先前的同步报文准确的发送时间 t_{m1} 。

假设系统的网络延迟时间为 t_{Delay} ,从时钟和主时钟的偏移时间为 t_{Offset} ,则:

$$t_{s1} = t_{m1} + t_{Offset} + t_{Delay} \quad (1)$$

③ 从时钟向主时钟发送“延时请求报文”,延时请求报文的间隔是独立设置的。报文发送后由从时钟记下准确的发送时间 t_{s2} ,在主时钟端记录准确的接收时间 t_{m2} 。

④ 主时钟向从时钟返回一个“延时响应报文”,此报文包含先前的“延时请求报文”的准确接收时间 t_{m2} ,从时钟利用这个接收时间和由它记录的发送时间 t_{s2} ,可以得到:

$$t_{m2} = t_{s2} - t_{Offset} + t_{Delay} \quad (2)$$

根据网络链路的对称性,由式(1)和式(2)可以计算出报文传输的平均延迟时间:

$$t_{Delay} = \frac{t_{s1} - t_{m1} + t_{m2} - t_{s2}}{2} \quad (3)$$

⑤ 重复步骤①,由主时钟发送新的同步报文,可

以在从时钟端得到新的同步报文的接收时间 t_{s3} 。

⑥ 重复步骤②,由主时钟发送新的跟随报文,可以得到新的同步报文的准确发送时间 t_{m3} 。

则主从时钟的偏移为:

$$t_{Offset} = t_{s3} - t_{m3} - t_{Delay} \quad (4)$$

根据求出的偏移 t_{Offset} 和延迟 t_{Delay} ,如果目前的从时钟时间为 t_{sOld} ,则可以计算出目前应该同步到的新的从时钟时间 t_{sNew} :

$$t_{sNew} = t_{sOld} - t_{Offset} - t_{Delay} \quad (5)$$

据此,在接下来的一个同步周期内,就可以实现从时钟和主时钟的同步。以上是整个时钟同步过程的具体步骤。

在实际的分布式应用中,同步报文和跟随报文的发送是按照定义的时间间隔周期性发送的,延迟请求报文和延迟应答报文发送间隔可以是不规则的,其测量间隔时间缺省值是 4~60 s 之间的随机值,比偏移值测量间隔时间要大^[5],这样做可以减轻网络尤其是终端设备的负荷。

2.2 速率校正

IEEE 1588 采用时间分布机制和时间调度概念,系统内的从时钟设备可使用普通振荡器,通过软件调度与主时钟保持同步。

速率校正通过测量每个报文从主时钟开始与到达从时钟之间的差异,计算出从时钟频率相对于主时钟频率的一个基本比值来校正频率差。由于报文延时偏差可能很明显,有可能使任何单一的速率测量有很大的不精确性,因此,可以利用平均路径延时测量来检测短时报文,并使用这些报文检测从时钟对主时钟的频率偏移^[6]。

为避免从时钟的抖动和振荡,维持时钟的单调性,精确时钟同步协议 PTP 没有立即用前面得到的时钟偏差直接调整本地时钟,而是把对 PTP 节点本地时钟的校正处理看作一个锁相环,通过一个环路滤波器转换为一个对可变频率振荡器的控制量来控制振荡器的振荡频率,从而间接调整系统的从时钟^[7]。

3 报文时间戳的生成

PTP 可以由软件记录报文的发送时间戳和接收时间戳,也可以由硬件记录。如图 4 所示,PTP 支持在物理链路层 A、驱动程序层 B 和应用软件层 C 处检测和记录报文发送和接收的时间戳。越靠近物理链路层,同步精度越高,越靠近应用软件层,同步精度越低。早期的网络时间协议 NTP 是只有软件控制的同步协议,只在应用软件层打上时间戳,因此同步的精度不

高,而 PTP 为了提高系统的时钟同步精度,既使用软件,也使用硬件和软件配合,能获得更精确的定时同步。PTP 支持硬件辅助在物理链路层检测和记录报文的时间,从而把报文从驱动程序到应用软件处理的不确定时间延迟排除在系统精度的分析之外^[8]。



图 4 报文时间戳的检测和记录

Fig. 4 Detection and record of message timestamp

目前已有多种芯片和微控制器支持 PTP 时间戳硬件实现,常见的有美国国家半导体的 DP83640 芯片、Silicon Labs 公司生产的 Si5315 芯片和飞思卡尔的 ColdFire MPU 等,也可以采用 FPGA 设计实现。当采用硬件辅助检测和记录 PTP 报文的时间戳时,即在以太网物理层 PHY 中放置这些器件,一旦线路上有 PTP 包,即被这些器件读取^[9]。因此,支持 PTP 时间戳的硬件功能实现是保证时间同步精度的关键器件。采用此设计方法的一个优点是仅需替换掉以太网物理层并增加 IEEE 1588 PTP 软件,便可将这种解决方案添加到已有的产品设计中,从而避免转换到新的处理器系列或开发辅助 FPGA 的复杂过程。

采用 DP83640 芯片实现以太网时钟同步的系统结构图如图 5 所示。

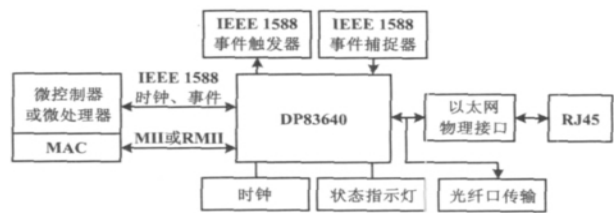


图 5 系统结构图

Fig. 5 Structure of the system

当采用 PTP 时,本地时钟相对于主时钟的同步时钟精度可由 PTP 的标准差或称均方差表示,PTP 的标准差基于 Allan 标准差,Allan 标准差 $\sigma_y(\tau)$ 的估计如式 6 所示:

$$\sigma_y(\tau) = \left[\frac{1}{2(N-2)\tau^2} \sum_{k=1}^{N-2} (x_{k+2} - 2x_{k+1} + x_k)^2 \right]^{\frac{1}{2}} \quad (6)$$

式中 x_k, x_{k+1}, x_{k+2} 为对应时间 $t_k, t_k + \tau, t_k + 2\tau$ 的样本测量值, τ 为样本测量的时间间隔, N 为样本总数。Allan 标准差是对时基振荡器频率变化的二阶统计,

PTP 标准差则是对测量时钟和本地参考时钟的偏差统计,PTP 方差定义为:

$$\sigma_{PTP}^2 = \tau^2 \times \frac{1}{3} \sigma_y^2 \quad (7)$$

由式(6)和式(7)可得 PTP 的标准差无偏估计的计算如式(8)所示:

$$\sigma_{PTP} = \left[\frac{1}{6(N-2)} \sum_{k=1}^{N-2} (x_{k+2} - 2x_{k+1} + x_k)^2 \right]^{\frac{1}{2}} \quad (8)$$

式中 x_k, x_{k+1}, x_{k+2} 为对应时间 $t_k, t_k + \tau, t_k + 2\tau$ 的测量值; τ 为样本测量的时间间隔; N 为样本总数。

测试系统时钟同步精度的方法有多种,最常用的是通过示波器观察秒脉冲信号 (pulse per second, PPS) 的波形^[10]。

为了使测试的精度能够满足实际的工程应用,对不同情况下的以太网网络负载分别进行了相应的测试,测试系统的组成如图 6 所示。



图 6 同步精度测试系统的组成

Fig. 6 Compositions of the test system for synchronization accuracy

设置主时钟和从时钟在某一频率点产生一个时钟输出,然后比较这两路时钟信号。采用秒脉冲测量是由于秒脉冲信号一般从数字输出中产生,虽然会对同步结果增添额外的误差,但是附加的误差仅会影响数字输入和输出,而不是同步时钟本身,因此不会包括在同步测量中^[11]。时钟同步精度秒脉冲测试结果如表 1 所示。

表 1 测试结果

Tab. 1 Test result

网络流量	利用率 /%	标准差 /ns
20		13.9
50		15.7
80		28.0

由表 1 可知,当网络负载在 80% 时,从时钟和主时钟的同步精度在 28.0 ns 的标准差内,完全能够满足分布式控制系统的时钟需求。

4 结束语

随着分布式测量和控制系统控制节点规模的不断增大,节点间的距离分散化和数据传输的网络化,使节点间的时钟同步变得愈加重要。依靠传统的专用线缆

的方式同步系统时钟,只会增加系统的复杂性、降低系统的可靠性。

IEEE 1588 精确时钟同步协议的出现,为分布式测控系统的时钟同步提供了一种切实可行的途径。由于 PTP 的应用建立在系统的通信网络,不局限于以太网上^[12],不需要为时钟的传递布置特别的网络,只需在原有网络上添加时钟同步报文。这些报文只占用少许的网络资源,它们和控制数据包或其他信息包共享网络,具有较低的网络开销,在不需要太多资源的情况下,就能实现微秒级的高精度时钟同步^[13]。

IEEE 1588 标准的精确时钟协议,代表了控制网络时钟同步技术发展的新趋势,值得进一步的研究和应用推广。

参考文献

- [1] IEEE Instrumentation and Measurement Society. IEEE Std 1588-2008 IEEE standard for a precision clock synchronization protocol for networked measurement and control systems [S]. New York, NY 10016-5997, IEEE-SA Standards Board 2008.
- [2] International Electrotechnical Commission. IEC 61588-2009 Precision clock synchronization protocol for networked measurement and control systems [S]. IEC under Licence from IEEE, ISBN 2-8316-1026-3, 2009.
- [3] Mahmood A, Gaderer G. Timestamping for IEEE 1588 based clock synchronization in wireless LAN [C]//ISPCS 2009 International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication. Italy 2009 63-68.
- [4] Rodrigues S. IEEE 1588 and synchronous Ethernet in telecom [C]//ISPCS 2007 International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication. Gaderer 2007 138-142.
- [5] 魏丰, 孙文杰. IEEE 1588 协议时钟同步报文的精确时间标记方法研究 [J]. 仪器仪表学报 2009 1()162-169.
- [6] 王颀, 王平, 向敏. EPA 网络控制系统时钟同步主时钟控制算法研究 [J]. 系统仿真学报 2009 1()146-151, 156.
- [7] 朱宝, 杨顺昆. 实时分布式仿真测试平台时钟同步算法 [J]. 计算机工程 2010 24()231-232, 235.
- [8] 王波, 吕俊伟, 于振涛. 基于 D-S 的时钟同步竞争安全算法 [J]. 电讯技术 2011 1()61-67.
- [9] 李学桥, 陈园, 梁爽. 基于 IEEE1588 协议的精确时钟同步算法改进 [J]. 计算机工程与科学 2011 2()42-45.
- [10] 章涵, 冯冬芹, 褚健. 基于透明传输的数控系统时钟同步算法 [J]. 计算机集成制造系统 2010 4()873-877.
- [11] 鲁立, 冯冬芹, 褚健, 等. 传输时延和时钟同步对以太网控制系统的影响 [J]. 控制理论与应用 2010 6()793-798.
- [12] 于会游, 周春晖, 许希斌, 等. 分组通信网络时钟同步研究及性能仿真 [J]. 计算机仿真 2009 5()173-178.
- [13] 蔡君, 范馨月, 周非. 分组网络时钟同步算法的研究 [J]. 电信科学 2010 2()227-231.

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 3.0 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](#)
70. [CAN 总线的浅析 CANopen 协议](#)
71. [基于 CANopen 协议实现多电机系统实时控制](#)
72. [以太网时钟同步协议的研究](#)

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](#)
52. [基于 Tilcon 航海标绘台界面设计](#)
53. [基于 Tornado 和 Tilcon 的嵌入式 GIS 图形编辑软件的开发](#)

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 \$\mu\$ C-OS II 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 COM Express 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 Nios II 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](#)
10. [基于小波变换与偏微分方程的图像分解及边缘检测](#)
11. [基于图像能量的布匹瑕疵检测方法](#)
12. [基于 OpenCL 的拉普拉斯图像增强算法优化研究](#)
13. [异构平台上基于 OpenCL 的 FFT 实现与优化](#)

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. [基于 FPGA 和 DSP 的 DBF 实现](#)
3. [高速浮点运算单元的 FPGA 实现](#)
4. [DLMS 算法的脉动阵结构设计及 FPGA 实现](#)
5. [一种基于 FPGA 的 3DES 加密算法实现](#)
- 6.