

## GPI接口的 FPG实现

陈立超 杨小雪

西南交通大学 四川 成都 610031)

摘要 介绍了 GPI控制器的总体结构 ,论述了基于 FPG的 GPI接口设计 给出了源方挂钩接口功能的状态图和完整的程序代码 同时对该程序代码进行了编译、综合和仿真。

关键词 : GPI接口 ; FPGA源方挂钩

### 0引言

在自动测试领域中 , GPIB (General Purpose Interface Bus)通用接口总线仪器系统是现代测量技术与计算机技术相结合的一个范例 是一种通过外总线互联的计算机测试系统 也是测试仪器常用的接口方式。通过 GPI组建自动测试系统的设计比较方便而且费用低廉。 GPI控制芯片是自动测试系统中的关键芯片 而此类芯片的量产受到专业用途的限制 ,目前国外只有德州仪器 (TI) NATIONAL INSTRUMENT (NI)和 NEC家公司生产基于 IEEE-488协议的 GPI控制器芯片 ,而且价格昂贵 购买不便。因此 , GPI接口的 FPG实现就具有很大的实用价值。本文论述的是采用 Verilog HDL语言来设计基于 IEEE488协议的 IP Core然后将用户逻辑与此 Core集成在一片 FPG从而实现对 GPI接口功能的一种新方法。

### 1 GPI控制器的总体结构

用 FPG实现 GPI控制器可采用 ISP系统可编程的设计方法。先把整个待设计系统划分为若干基本功能模块 其中包括基本组合逻辑电路以及复杂的同步状态机设计。

在基本单元实现方法上 ,可采用语言描述方式完成基本模块的设计 这不仅方便将来跨开发环境的移植 而且在功能调试以及对局部修改都能带来很大的方便。其内部逻辑划分可基本参照 公司的 TMS99芯片来进行。图 所示是

GPI控制器的体系结构图。

在完成基本模块以后 即可对每个模块生成图形化的符号 (Symbol)在组建整个系统的时候 可以用类似画电路图的方法直接对这些符号进行调用 从而设计相对比较直观而且对整个系统级架构能够全面把握的系统。在完成整个内部电路的“软件”设计以后 便可把该“软件核”(CORE)固化到具体选定的 FPG 最后利用 JTAG接口下载。

### 2系统分解

根据 GPI控制器的内部逻辑框图 可以把整个系统划分为几个子系统。其中包括和微处理器接口的读写电路 与 GPIB线通信的接口电路 ,以及内部寄存器。对每个子系统的设计也可以采用向下分解为模块级的设计方法 这样 在调试和组件系统时 不仅具有很强的灵活性 而且可以方便代码的调试 也利于代码的重利用。

与微处理器接口的读写电路设计可以利用组合逻辑电路设计方法来实现 其基本模块有译码电路 读写电路等在设计时 只要对其基本功能深刻理解 就很容易实现。

与 GPIB线通信的子系统设计不仅需要对其对 IEEE488协议有深刻的认识 而且在接口功能子集的选择上也有一定的要求。为了让各接口功能子集能协调一致的工作 必须采用同步状态机。

系统中的内部寄存器由 寄存器组成 其中包括只读寄存器和只写寄存器。它们是完成微处理器端到 GPI接口功能端的数据桥梁 其中一

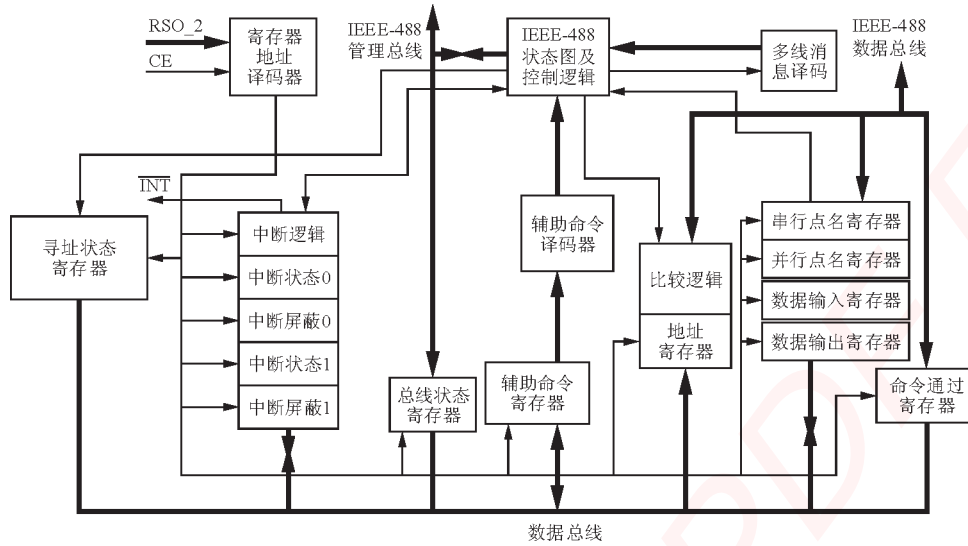


图 1 GPIB 控制器体系结构简图

些寄存器还赋予了 GPIB 控制器的若干片内附加功能。

在整个设计中，对接口功能的设计是设计的核心。接口功能的完成必须依赖于各子接口功能的实现。在实现子接口功能的时候，按照 IEEE488 协议的要求，可以把各子接口划分为若干个相关状态。这样，在用 Verilog HDL 实现这些相互作用且有影响的状态跃迁的时候，就不可避免地需要引入状态机设计。

### 3 接口功能的实现

#### 3.1 子接口的选取

一个接口功能可提供一种基本操作手段。器件能通过它来接收、处理和发送消息。在 IEEE488-1976 标准中规定了十种接口功能：源方挂钩、受方挂钩、讲者、听者、控者、并行查询、服务请求、远控本控、器件清除和器件触发。每种接口功能按照特殊的规约来行事。一个接口功能可以有一个或多个容许的子集。其中源方挂钩接口功能保证多项消息的正确传输。受方挂钩接口功能主要用于保证正确地接受远地消息。讲者接口功能可让器件在接口上把数据发送到其他一些器件中去。听者接口功能让器件在接口上接受来自于其他器件的数据。本控就是本地控制（面板或后背板）。为了保持状态机之间相互的状态挂钩的协调一致，设计时应采用同步状态机。以在时钟信号的触发下，完成各个状态之间

的转移。

每一个接口功能由一组或多组相互联系而又相互独立的状态来定义。在一个时刻，一组互斥又互斥的状态中必须有一个而且也仅有一个状态起作用。在 IEEE488 协议中对接口功能的每一个状态还做了两方面的规定：

第一，当某个状态起作用时，在接口上可以或必须发送消息；

第二，在什么条件下接口功能必须脱离某一状态而进入同一组状态的另一个状态；

这些因素决定了状态的处理能力。可使用接口功能状态图给出全部状态以及从一个状态变迁到另一个状态的一切可能的途径。

#### 3.2 源方挂钩接口功能

源方挂钩（SIF）功能主要是赋予器件确保多线消息正确传递的能力。源方挂钩功能与一个至多个受方挂钩功能，分别含于各个仪器之中。它们之间的链锁挂钩序列，可保证每一个多线消息的异步传递。SIF 功能控制着多线消息比特传递的开始和结束。它接收由受方器件传来的 RFD/DAC 消息，并产生 DA 消息来实现对数据比特传递的控制。其状态图如图 2 所示。

##### (1) 状态图

源方空闲态（SIFS）：在电源接通时使 SIF 能进入 SIFS。在这一态，SIF 不参与挂钩循环，而且没有可供使用的消息比特。

源方产生态（SGNS）：用于在源方产生器件

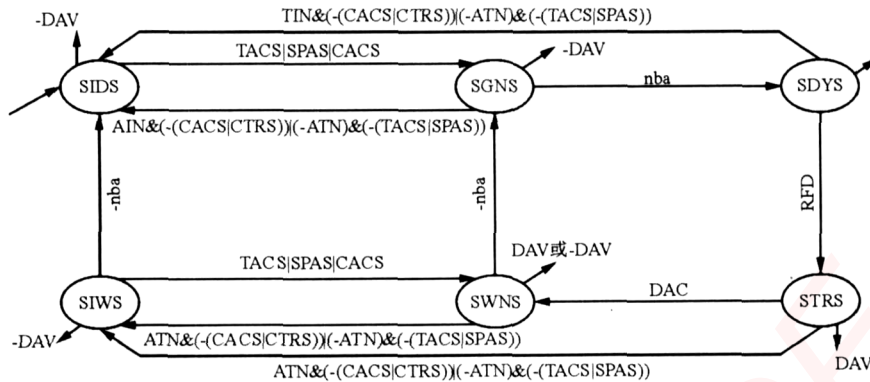


图2 SH功能状态图

产生一个新的消息比特 以使 功能正等待的新消息变成可供使用的消息。 D 消息应被发送为假态 ( - ) 器件可以改变正在通过讲者功能 ( T 真 ) 或控者功能 ( C 真 ) 发送的多线消息。

在源方延迟态 ( S D ) 功能等待在 S G N S 跃迁到 S S G 由此可知 该源方挂钩功能态改变了的消息比特在母线上稳定下来。 功能还必须停留在 S 态等待 S 直到参加挂钩的 A H ( A 假 T 方试。 =

在源方传递态 ( S T ) 功能向受方挂钩功能表明器件正在继续发送有效的消息比特 此时 , D 消息必须被发送为真消息。不言而喻 , 这时的器件不能改变或结束正在传递的多线消息。

在源方等待新循环态 ( S W ) 功能等待器件开始一轮新的消息循环。此时 D 消息可发送为真 , 也可以发送为假 器件可以改变 D 线上的多线消息。

源方闲等态 ( S W ) 功能对外部比特传递过程不起作用 但在内部等待器件开始一次新的消息循环产生过程中仍起作用。这个 S I W 态容许消息比特传递序列被中断而不丢失接口母线上的数据 同时 器件可以继续准备下一个消息比特的产生循环。此态 D 消息必须发送成被动假 而且器件可以改变将要被传送的多线消息。

### (表语)

在这组状态图中 箭头用来表示一个接口功能中各状态之间的一切允许的变迁。每一项变迁都由一个表语来限定 表语之间可以为真 , 也可

以为假。如果限定变迁到另一个状态的一切表语均为假 则接口功能必须停留当前的状态。并在若且仅若表语之一变为真之后的任何时间内进入。

通过状态表语 T A C S | 可以看到 S | C A

设计包括命令工作方式 ( A 真 T 及数据工作 ( 基于 ) V e r 的状态机实现 H D L

根据以上分析 状态机的组成一般分由组合逻辑和时序逻辑 这样 在描述状态机的时候 , 只要对接口电路的状态图分析清楚 即可用相同的语言进行描述 即代码结构都相同。值得注意的是如何把 I E 对接口功能描述转化为相应的状态图。

鉴于 S 模块十分重要 而且作为其它状态机的标准代码模板 本文特给出该状态机的代码 :

```

module fSH ( S I D S , S G
S I W S , D A V , clk , rst ,
S P A S , n b 对模块信号进行定义 R F D )
input clk , rst
R F D
output t D A V
output S I D S , S G ; N S , S D
reg S I D S , S G N S ; , S D Y S ,
reg ; r D A V
parameter [ 5 : 0 ]
s S I D S 采用6位热码编码格式 0 0 1 , / /
s S G N S = 6 ' b 0 0 0 0 1 0 , s S
6 ' b 0 0 1 0 0 0 , s S W N S = 6 '
    
```

```

sSIWS= 6'b100000;
reg [5:0] state, next;
assign DAV = ( SGNS| SDYS| STRS| SWNS| SIWS
) ? rDAV : 1'bz;
always@ ( negedge clk or posedge rst )时序逻辑
begin
    if ( rst ) state = sSIDS;
    else state = next;
end
always@ ( state or ATN or CACS or CTRS or TACS
or SPAS or nba or DAC or RFD )组合逻辑
begin
    SIDS= 0; SGNS= 0; SDYS= 0; STRS= 0;
SWNS= 0; SIWS= 0; rDAV = 0;
case ( state)
sSIDS:
    begin
        if ( TACS| SPAS| CACS)状态表语
            next = sSGNS;
        else next = sSIDS;
        SIDS= 1;
        rDAV = 1; //rDAV = 0; passive DAV_
message
    end
sSGNS:
    begin
        if ( nba ) next = sSDYS;
        else if ( ( ATN & { ( CACS| CTRS ) } ) |
( { ATN } & { ( TACS| SPAS ) } ) )
            next = sSIDS; 对 ATN的响应时间
        else next = sSGNS;
        SGNS= 1;
        rDAV = 1; // rDAV = 0; DAV_ message
    end
sSDYS:
    begin
        if ( RFD ) // T1 > 2us总线消息稳定时间
            ##2000 next = sSTRS;
        next = sSTRS;
        else if ( ( ATN & { ( CACS| CTRS ) } ) |
( { ATN } & { ( TACS| SPAS ) } ) )
            next = sSIDS; // @t2
    else next = sSDYS;
        SDYS= 1;
        rDAV = 1; //rDAV = 0;
    end
sSTRS:
    begin
        if ( DAC ) next = sSWNS;
        else if ( ( ATN & { ( CACS| CTRS ) } ) |
( { ATN } & { ( TACS| SPAS ) } ) )
            next = sSIWS;
        else next = sSTRS;
        STRS= 1;
        rDAV = 0; // DAV Truer
        DAV = 1;
    end
sSWNS:
    begin
        if ( ( ATN & { ( CACS| CTRS ) } ) | ( (
~ ATN ) & { ( TACS| SPAS ) } ) )
            next = sSIWS;
        else if ( ! nba ) next = sSGNS;
        else next = sSWNS;
        SWNS= 1; // DAV can be any state
    end
sSIWS:
    begin
        if ( ! nba ) next = sSIDS;
        else if ( TACS| SPAS| CACS ) next = sS
WNS;
        else next = sSIWS;
        SIWS= 1;
        rDAV = 1; //rDAV = 0;
    end
default//机进入默认状态
    begin
        next = sSIDS;
    end
endcase
end
endmodule

```

在以上代码编译通过之后，便可以通过 Synplify 进行综合。这样就可以分析具体资源的利用以及逻辑结构是否满足设计需求。在进一步优化的基础上生成的符号（Symbol）及其互连方式如图 3 所示。

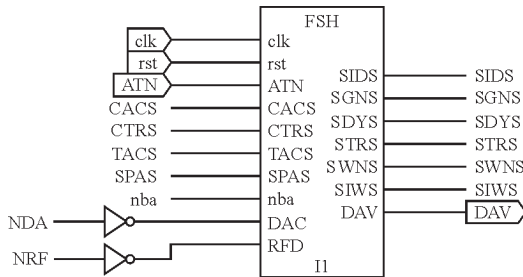


图 3 符号以及互连方式

以上符号连接是在系统结构中的连接方式。在用 MAXplus 仿真时，可以仅对模块进行仿真。此后，在对系统级进行仿真时，只有在所有模块完成的基础上才可实现。该源挂钩模块的器件受命为讲者时，其数据工作方式中的仿真波形如图 4 所示。

其它接口功能的状态机实现与 S 功能类似。将每个接口功能的状态机实现以后，即可生成便于互连的符号模块，各个模块连接起来就组成了系统的接口主状态机模块。其中状态机子集模块所需的信号，有一些来自状态交连，另外一些则来自其他模块。

### 3.4 接口的 FPGA 实现

在完成基本模块以后，就可以用类似画电路图的方法直接对这些模块符号进行调用。然后通过 MAXplus 进行综合、仿真和在线调试。最后，在完成整个内部电路的软件设计以后，就可以把该“软件核”（Core）固化到具体选定的 FPGA 芯片上，本文选用 ALTERA 公司的 ACEX1K30QC208 芯片来设计。

### 4 结束语

在测试仪器的研究中发现，GPIB 接口往往不需要 IEEE488 的全部接口功能，很多仪器只需要听、讲功能即可。如果采用 TMS99 系列的芯片控者、并行查询功能都用不到，这样既会造成资源与功能的浪费，又会延长开发周期。而设计基于 FPGA 芯片的专用 IP Core 仅拥有自主知识产权，而且可以通过与计算机接口（如 ISA）与计算机直接相连，最后装入驱动程序即可工作。这对测试仪器开发有一定的借鉴作用。事实上，无论是采用 PCI、USB 还是单片机控制，只需将用户逻辑与符合 IEEE488 协议的 IP Core 集成在一片 FPGA 即可实现测试仪器的 GPIB 接口功能，而且移植方便灵活。故可大大缩短开发周期，降低开发成本。

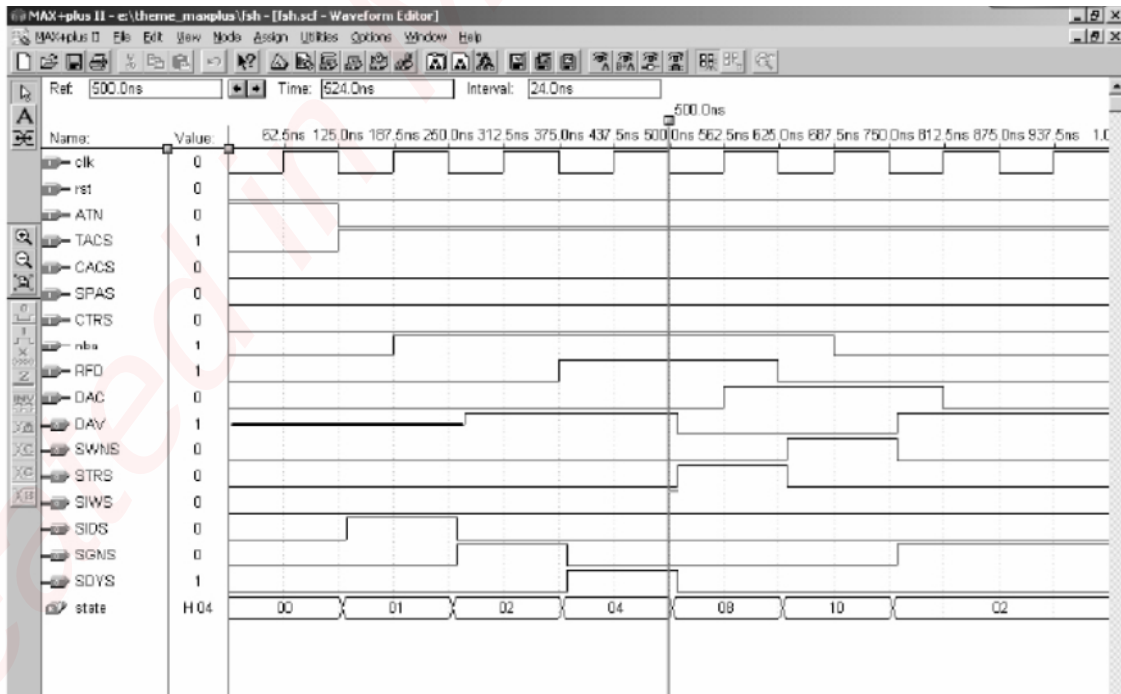


图 4 数据工作方式下的 S 功能时序仿真图

# 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. [基于磁盘阵列引擎的 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](#)
73. [基于 CANopen 的列车通信网络实现研究](#)
74. [基于 SJA1000 的 CAN 总线智能控制系统设计](#)
75. [基于 CANopen 的运动控制单元的设计](#)
76. [基于 STM32F107VC 的 IEEE 1588 精密时钟同步分析与实现](#)

77. [分布式控制系统精确时钟同步技术](#)
78. [基于 IEEE 1588 的时钟同步技术在分布式系统中应用](#)
79. [基于 SJA1000 的 CAN 总线通讯模块的实现](#)
80. [嵌入式设备的精确时钟同步技术的研究与实现](#)
81. [基于 SJA1000 的 CAN 网桥设计](#)
82. [基于 CAN 总线分布式温室监控系统的设计与实现](#)
83. [基于 DSP 的 CANopen 通讯协议的实现](#)
84. [基于 PCI9656 控制芯片的高速网卡 DMA 设计](#)
85. [基于以太网及串口的数据采集模块设计](#)
86. [MVB1 类设备控制器的 FPGA 设计](#)
87. [MVB 接口彩色液晶显示诊断单元的显示应用软件设计](#)
88. [IPv6 新型套接字的网络编程剖析](#)
89. [基于规则的 IPv4 源程序到 IPv6 源程序的移植方法](#)
90. [MVB 网络接口单元的 SOC 解决方案](#)
91. [基于 IPSec 协议的 IPv6 安全研究](#)
92. [具有 VME 总线的车载安全计算机 MVB 通信板卡](#)
93. [SD 卡的传输协议和读写程序](#)
94. [基于 SCTP 的 TLS 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](#)
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](#)
54. [VxWorks 环境下内存文件系统的应用](#)
55. [VxWorks 下的多重定时器设计](#)
56. [Freescale 的 MPC8641D 的 VxWorks BSP](#)
57. [VxWorks 实验五\[时间片轮转调度\]](#)
58. [解决 VmWare 下下载大型工程.out 出现 WTX Error 0x100de 的问题](#)
59. [基于 VxWorks 系统的 MiniGUI 图形界面开发](#)

60. [VxWorks BSP 开发中的 PCI 配置方法](#)
61. [VxWorks 在 S3C2410 上的 BSP 设计](#)
62. [VxWorks 操作系统中 PCI 总线驱动程序的设计与实现](#)
63. [VxWorks 概述](#)
64. [基于 AT91RM9200 的 VxWorks END 网络驱动开发](#)
65. [基于 EBD9200 的 VxWorks BSP 设计和实现](#)
66. [基于 VxWorks 的 BSP 技术分析](#)
67. [ARM LPC2210 的 VxWorks BSP 源码](#)
68. [基于 LPC2210 的 VxWorks BSP 移植](#)
69. [基于 VxWorks 平台的 SCTP 协议软件设计实现](#)
70. [VxWorks 快速启动的实现方法\[上电到应用程序 1 秒\]](#)

## 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 C++ 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](#)
52. [Linux 基于 W83697 和 W83977 的 UART 串口驱动开发文档](#)
53. [基于 AT91RM9200 的嵌入式 Linux 系统的移植与实现](#)
54. [路由信息协议在 Linux 平台上的实现](#)
55. [Linux 下 IPv6 高级路由器的实现](#)

## 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](#)
29. [Windows CENET 下 ADC 驱动开发设计](#)
30. [Windows CE 下 USB 设备流驱动开发与设计](#)
31. [Windows 驱动程序设计](#)
32. [基于 Windows CE 的 GPS 应用](#)
33. [基于 Windows CE 下大像素图像分块显示算法的研究](#)
34. [基于 Windows CE 的数控软件开发与实现](#)

## 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 I/O 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](#)
31. [基于 MPC850 的多功能通信管理器](#)
32. [基于 MPC8640D 处理系统的技术研究](#)
33. [基于双核 MPC8641D 处理器的计算机模块设计](#)
34. [基于 MPC8641D 处理器的对称多处理技术研究](#)

## 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](#)
36. [UBoot 在 AT91RM9200 上的移植简析](#)
37. [基于工控级 AT91RM9200 开发板的 UBoot 移植分析](#)

## 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 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 技术研究与实现](#)
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](#)
41. [EFI-Tiano 环境下的 AES 算法应用模型](#)
42. [EFI 及其安全性研究](#)

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](#)
14. [数据结构考题 - 第 4 章 串](#)
15. [数据结构考题 - 第 4 章 串答案](#)
16. [用 IPv6 编程接口实现有连接通信的方法](#)

## 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. [可编程 FIR 滤波器的 FPGA 实现](#)
7. [基于 FPGA 的 AES 加密算法的高速实现](#)
8. [基于 FPGA 的精确时钟同步方法](#)
9. [应用分布式算法在 FPGA 平台实现 FIR 低通滤波器](#)
10. [流水线技术在用 FPGA 实现高速 DSP 运算中的应用](#)
11. [基于 FPGA 的 CAN 总线通信节点设计](#)
12. [基于 FPGA 的高速时钟数据恢复电路的实现](#)
13. [基于 FPGA 的高阶高速 FIR 滤波器设计与实现](#)
14. [基于 FPGA 高效实现 FIR 滤波器的研究](#)
15. [FPGA 的 VHDL 设计策略](#)
16. [用 FPGA 实现串口通信的设计](#)