

FPGA 的 VHDL 设计策略

罗 沈绪榜 高德远

(西北工业大学 航空微电子中心, 陕西 西安 710072)

摘要: 大规模的可编程逻辑器件已经显著改变了数字系统的设计过程, 并且 VHDL 语言在设计中的作用也日益显著。简要论述了关于 FPGA 的 VHDL 设计中一些注意事项, 提高电路描述的正确性, 从而提高 FPGA 设计的性能。

关键词: FPGA (现场可编程逻辑门阵列); VHDL (硬件描述语言); 可综合

中图分类号: TP33

文献标识码: A

文章编号: 1000-1220(2003)07-1194-03

VHDL Design Strategy in FPGA

LUO M in, SHEN Xu-bang, GAO De-yuan

(Aviation Microelectronic Center Northwestern Polytechnical University, Xi'an 710072, China)

Abstract: High density PLD has changed the design process of digital system evidently, and VHDL is more and more important in the design process. This paper discusses some points of VHDL in FPGA to improve the validity of the circuit, and so we can improve the performance of the FPGA design.

Key words: FPGA; VHDL; Synthesizable

1 概述

目前, 可编程逻辑器件(PLD)的密度、速度和 I/O 接口已经可以满足大多数的数字 ASIC 的设计要求, 因而得到了越来越广泛的使用。同时, 所涉及的门数已经从几百门增加到几十万门, 使得原先设计所使用的逻辑图输入、真值表和卡诺图的设计方法已经远远不能满足数字系统的复杂性要求。采用硬件描述语言(Hardware Description Language)作为设计输入的方法是近年来 ASIC 设计的最新的、最先进的设计方法。

目前, 流行的 FPGA 设计方式是: 用 VHDL 语言作为硬件输入描述, 对 HDL 硬件描述通过采用综合器综合使硬件语言描述映射到逻辑电路上, 然后再采用 FPGA 厂商提供的工具进行布局布线将综合产生的逻辑电路真正对应到具体 FPGA 结构上这样一套流程。

在这样的设计流程中, 综合以及后面的布局布线工作都是基于 HDL 描述上进行的, 因此硬件语言描述的好坏就直接决定了最后的综合后的电路的优劣。同时, 在 FPGA 设计中, 仅仅只是利用硬件描述语言和综合技术是不够的。如果想要最终器件在性能和面积上都达到优化, 就必须熟悉器件的体系结构, 然后针对该器件进行 VHDL 设计。

2 FPGA 的特点

FPGA 通常被认为是 ASIC 实现的一种替代手段。一般 ASIC 包括三种, 既全定制、半定制(含标准单元和门阵列)以及可编程器件。对于前两种, 需要支付不可重复使用的工程费用 NRE (Non recurring Engineering), 主要用于芯片的流片、

中测、分析的工程开销, 一次费用一般在 1 万至数万美元以上。如果一次不成功, 返工, 甚至多次返工, NRE 费用将要上升。成本高, 风险大, 而通常对每个 ASIC 品种的需求量往往不大, NRE 费用分摊到每个产品上价太高, 用户无法接受。而对于可编程器件 PLD (Programmable Logic Device) 正是可以解决上述问题的新型 ASIC, PLD 以其操作灵活、使用方便、开发迅速、投资风险小等突出优点, 特别适合于产品开发初期、科研样品研制或小批量的产品。FPGA 是一种新型的 PLD, 其除了具有 PLD 的优点外, 其规模比一般的 PLD 的规模大。目前, Xilinx 推出的 XC4025 可以达到 25000 门的规模, Altera 公司的 FLEX10K100 系列芯片可达到十万门的规模, 完全可以满足用户的一般设计需要。

FPGA 的主要特点是: 寄存器数目多, 采用查找表计数, 适合时序逻辑设计。但是互连复杂, 由于互连采用开关矩阵, 因而使得延时估计往往不十分准确。

FPGA 也有其自身的局限性, 其一就是器件规模的限制, 其二就是单元延迟比较大。所以, 在设计者选定某一 FPGA 器件后, 要求设计者对器件的结构、性能作深入的了解, 在体系结构设计时, 就必须考虑到器件本身的结构及性能, 尽可能使设计的结构满足器件本身的要求。这样就增加了设计的难度。

离开对 FPGA 结构的详细了解, 设计人员就不可能优化设计。因而设计人员必须了解 FPGA 器件的特性和限制, 熟悉 FPGA 的结构。

在了解 FPGA 结构特点的基础上, 就可以利用 VHDL 语言描写出高效的电路描述实现性能优化的电路。

3 VHDL 可综合语言模型

在数字电路设计中, 由于输入的到达时间不完全一样, 可能会引起竞争冒险现象, 因此, 如果在设计中可以避免这种情况的发生, 那么就可以大大提高设计的成功率。为了避免这种情况, 我们在设计中采用 RTL 描述, 也就是寄存器传输级描述。就是在整个电路设计中, 以寄存器作为主要对象, 在寄存器之间插入组合逻辑。如图 1 所示:

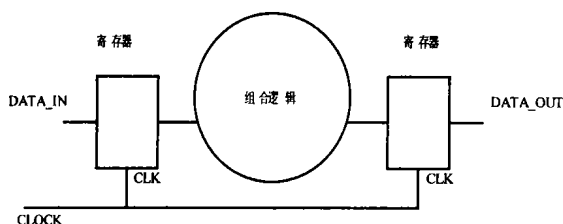


图 1 RTL 电路基本单元

Fig 1 The basic cell in RTL circuit

在具体的电路设计中, 应该尽可能将所有输入采用寄存器输入, 同时将所有输出采用寄存器输出。这样的设计, 使得所有的设计模块都是以寄存器为边界, 有利于综合器进行综合优化, 并便于进行静态时序分析。

3.1 时钟的使用

在 VHDL 语言中, 对于时钟的描述:

`wait until clk 'event and clk = '1';` 或者是

`if clk 'event and clk = '0' then`

其中 `event` 表示时钟信号进行变化, `clk` 为 1 表示是上跳沿, `clk` 为 0 表示是下跳沿。在 VHDL 描述中如果出现了上述语句, 那么在对生成电路中一定会出现寄存器。一般来说, VHDL 语言中的时钟描述应该遵循以下原则:

- 1) 如果异步复位, 那么时钟语句可以出现在 `elsif` 中, 否则应该在进程的首句中出现;
- 2) 一个进程只能有一个时钟;
- 3) 如果在进程中的其他位置出现时钟语句, 那么就有可能产生不可综合的现象;
- 4) 在整个设计中, 如果能够用一个时钟就用一个时钟。在一个设计中使用多个时钟, 不利于综合器进行优化, 也不利于布局布线。同时, 对于关键路径的分析也会受到很大影响;
- 5) 在整个设计中, 尽量统一使用一个时钟的上升沿或者是下降沿, 这样的设计便于综合优化, 便于时序分析。同时, 不同沿的设计会带来关键路径增长的问题。

3.2 时序部件

在数字电路中, 时序部件有两种: 寄存器、锁存器。其中, 寄存器是时钟边沿触发的, 锁存器是时钟电平触发的。在 FPGA 设计中, 应该尽量采用寄存器设计, 而不是锁存器设计。原因在于: 锁存器无法阻断毛刺的传播, 对信号没有整形作用, 而毛刺很容易影响电路的正常工作, 导致电路的误操作。同时, 在 FPGA 中, 没有真正的锁存器部件, 所有的锁存器都

是用寄存器加上外围电路构成的, 这样一来, 电路面积就会变大。对于电路的综合优化来说, 目前的工具对于寄存器的优化作用要优于对锁存器的。因此, 在 VHDL 描述中, 应该尽量采用寄存器描述。

一般来说, 往往在 VHDL 描述中锁存器的产生是由于对信号量的不完全赋值引起的。例如:

```
if a = '1' then
    b <= c;
end if;
```

这样的赋值语句, 将导致一个锁存器的产生。当然, 这个电路的功能是正确的, 但是这样带来的结果是面积增大。因此, 如果这个电路关心的只是 `a = 1` 时候的情况, 那么就可以对 `a` 进行缺省赋值。

```
b <= '0';
if a = '1' then
    b <= c;
end if;
```

这样, 产生的就是一个组合逻辑。

如果在设计中, 锁存器无法避免, 那么应该用时钟沿来寄存。

```
wait until clk 'event and clk = '1';
if a = '1' then
    b <= c;
end if;
```

这样, 就产生一个寄存器, 避免的毛刺的传播。

3.3 同步和异步置位

存储部件在上电后状态是不确定的, 为了确定状态, 电路采用复位信号来设置初值。根据与时钟的关系, 分为同步复位和异步复位。同步复位只有在指定的时钟沿才发生作用, 而异步复位一旦有效就发生作用。如图 2 所示:

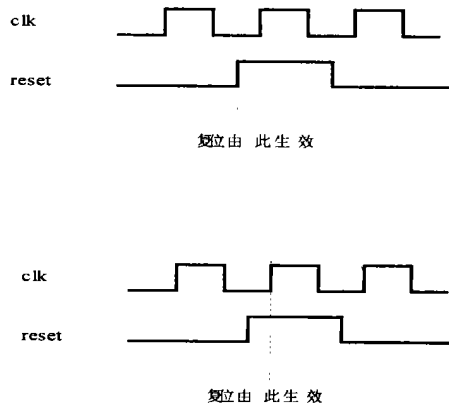


图 2 异步复位和同步复位

Fig 2 A synch reset and synch reset

由于异步复位容易受到毛刺影响, 因此在实际设计中, 除了将外部输入管脚信号作为异步复位信号外, 其他所有复位信号都作为同步复位信号。

4 针对 FPGA 的设计

上面是一些 VHDL 语言的设计基本规范,对于 FPGA 来说,根据其结构特点,还有一些注意事项

4.1 计数器设计

在 FPGA 设计中,最好使用 LFSR 计数器,而不是二进制计数器 原因是: LFSR 更加象移位寄存器,速度很快,同时,二进制计数器在翻转的时候,会产生毛刺,如果这个时候计数器的值被作为其他信号的输入,那么结果必将是错误的同时, LFSR 的扇入比较小,而二进制计数器的扇入比较大,虽然所用的寄存器数目二进制计数器比其他两种要少,但是从总的规模上来说,或者在性能上, LFSR 计数器要优于二进制计数器

下面实现了计数 8 的 LFSR 计数器和二进制计数器, VHDL 语言描述如下:

1) LFSR 计数器

```
if rst= '0' then
  cnt<= (others=> '1');
elsif clk'event and clk='0' then
  cnt<= lfsr_hign&cnt(7 downto 1);
end if;
lfsr_hign<= cnt(0) xor cnt(4) xor cnt(5) xor cnt(6)
```

2) 二进制计数器

```
if rst= '0' then
  cnt<= (others=> '0');
elsif clk'event and clk='0' then
  cnt<= cnt+ '1';
end if;
```

表 1 给出了两种计数器在 xilinx 器件 XC2S15VQ100 上实现的面积比较和最大速度

表 1 两种计数器的对比

Table 1 The compare of two kinds of counters

	LFSR 计数器	二进制计数器
SLICE 数目	4	4
SLICE 寄存器数目	8	8
4 输入 LUT 数目	1	0
等效门数	70	109
最快频率	264.90MHz	186.32MHz
最大线时延	1.576ns	1.409ns
最大组合路径时延	8.424ns	7.461ns

从表 1 可以得到, LFSR 计数器的等效面积要比二进制计数器的等效面积小 30%, 最高频率要高 40%. 因此, 对于某些只需要计数过程, 而不需要计数值的设计, 比如说计时器,

就完全可以使用 LFSR 计数器代替二进制计数器

4.2 多路选择器设计

多路选择器在 FPGA 中有两种实现方式, 一种是多选器, 一种是三态逻辑 VHDL 描述如下:

多选器

```
c<= a when (sel= '0') else b
```

三态逻辑

```
c<= a when (sel= '0') else 'z';
c<= b when (sel= '1') else 'z';
```

表 2 是 16 选 1 设计的面积报告 (使用器件为 XC2S30-5CS144)

表 2 两种不同实现方式多选器的面积、时序报告

Table 2 The area and timing reports of two MUXs based on two different

	三态逻辑	多选器
使用的 SLICE 数目	8	5
使用的 4 输入 LUT 数目	16	9
使用的 TBUF 数目	16	0
等效门数	147	72
最大组合延时	15.669ns	12.959ns
最大线延时	4.921ns	1.953ns

从表 2 来看, 发现无论是面积大小, 还是速度快慢, 用多选器实现的设计都要优于三态逻辑实现的设计. 因此, 一般来说, 在 FPGA 中都使用多选逻辑来实现多选器. 三态逻辑一般只是用在总线设计上

5 结论

表 1 和表 2 都是在特定 FPGA (Xilinx Spartan 2) 器件上实现的结果. 如果使用不同的 FPGA 或者是 CPLD 器件, 那么根据器件结构的不同, 很可能出现不同的结论. 这就需要设计者在设计之前了解所选器件的结构特点来确定自己的编码方式, 以实现面积和速度的最佳方案

VHDL 语言是提高设计速度的强有力工具, 通过掌握这门语言, 并且深入了解 FPGA 的结构特点, 又针对性的编程, 可以发挥器件最大的性能, 同时避免不必要的面积、速度甚至功耗上的损失

References

1 Steve Carison. Introduction to hdl-based design using vhdl [S] Synopsys NC, 1991
2 The programmable logic Data Book 2000 [Z] Xilinx NC, 2000

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](#)
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](#)

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. [Freescall 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 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](#)
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](#)

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](#)

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.