·开发研究与设计技术· 文章编号：1000—3428(2004)增刊—0596—03 文献标识码：A 中图分类号：TP 302.1

# 基于DSP系统的开放性设计

王国威，谢康林
(上海交通大学计算机科学与工程系，上海 200030)

**摘 要**：就如何增强基于DSP系统的开放性设计提出了两种方法：(1)主机可通过DSP对FPGA动态配置，实现硬件功能的更改；(2)主机通过PCI接口对DSP软件的动态加载，实现软件功能的升级和扩充。

**关键词**：动态加载；数字信号处理器；现场可编程阵列

# Opening Design Based on DSP System

WANG Guowei, XIE Kanglin
(Department of Computer Science & Engineering, Shanghai Jiaotong University, Shanghai 200030)

**【Abstract】** This paper introduces two means to enhance the openning design based on DSP system: One is the host can config the FPGA through DSP to change hardware logical funtion; The other is the host can load program to DSP through PCI interface to upgrade the software.

**【Key words】** Dynamic load; DSP; FPGA

## 1 概述

在日益激烈的市场竞争中，产品是否具有灵活的二次开发接口和高度的开放性已成为产品研发成败的关键因素，这就要求我们的产品在软硬件方面都具有高度的可编程性。然而，这几年超大规模、低成本现场可编程器件和高速数字信号处理器的出现，为系统的开放性设计提供了可能。大规模可编程器件可以实现信号的物理层(如数据和时钟的定时提取)和数据链路层中复杂协议的处理(如卷积码的译码)，数字信号处理器可以实现数据链路层中较为简单和高层应用协议的处理(如HDLC帧的透零处理、解扰和时隙提取等功能)。由于可编程器件支持系统编程，通过CPU加载不同的逻辑模块，就可以对不同的信号进行处理。数字信号处理器通过主机接口模式，加载新的软件模块，实现现有产品功能的升级。本文通过对Altera公司的可编程器件硬件逻辑和TI公司的数字信号处理器软件动态加载功能的实现来展示系统的开放性设计过程。

## 2 主机通过DSP对FPGA动态配置

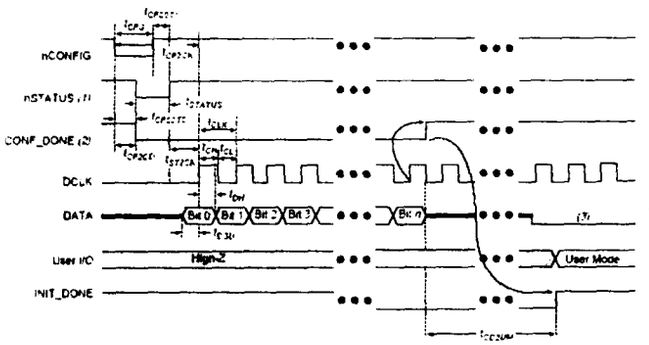


图1 PS配置方式时序图

Altera公司生产的具有在系统可编程功能的FPGA器件有Stratix、Cyclone和Apex等系列，其内部采用SRAM工艺，它的配置数据存储在SRAM中。由于SRAM的易失性，每次系统上电时，必须重新配置数据。它们的配置方式可分为PS(被动串行)、PPS(被动并行同步)、PPA(被动并行异步)、PSA(被动串行异步)和JTAG(Joint Test Action Group)等5种方式。这5种方式都可用CPU来配置。PS方式因其电路简单，对配置时钟的要求相对较低，而被广泛应用。EP20K300EQC240器件PS配置方式的时序图参见图1。

我们的配置方案是可编程器件EP20K300EQC240的配置管脚和TMS320C6205中的GPIO的管脚、读写控制信号连接，实际上就是把用于配置的控制和状态信号映射成DSP的IO寄存器，主机里的程序通过PCI总线对DSP里的这些寄存器进行读写，从而实现FPGA的配置。PS配置方式的逻辑图参见图2。

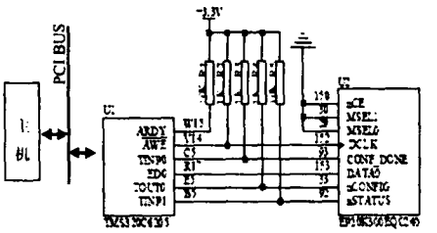


图2 PS配置方式逻辑图

被动串行工作过程：首先在nconfig脚产生一个宽度至少为8μs逻辑低信号，用于启动配置过程，接着在DCLK上升沿，将数据写入目标芯片。在配置过程中，系统需要实时监测，一旦出现错误，nSTATUS将被拉低，系统识别到这个信号后，报告主机配置失败。在配置数据全部正确地移入目标芯片内部后，CONF_DONE信号会跳变为高，此后，必须向DCLK提供几个周期的时钟(具体周期数与DCLK的频率有关)，确保目标芯片被正确初始化，进入用户工作模式。

配置文件可由Altera的Quartus II开发工具通过编译直接生成，也可由其它格式的文件(如.sof)转换而成，格式为.rbf，它是一种二进制文件。该文件包括所有的配置数据，每一字节在配置时最低位最先被装载。主机先读取这个二进制文件，放在内存中，然后启动配制过程。其配制程序(已在C++ Builder 6.0下调试通过)如下：

**作者简介**：王国威(1971－)，男，硕士生，主研方向：数据通信，网络存储技术；谢康林，教授、博导

**收稿日期**：2004-08-15



© 1995-2005 Tsinghua Tongfang Optical Disc Co., Ltd. All rights reserved.

```
extern "C" bool __export __pascal ConfigFPGA(HANDLE hDevice, CARD_INFO Card_info,void *Buf, DWORD Length)
{
  BYTE data;
  BYTE *databuf;
  WORD resault;
  databuf = (BYTE *)Buf;// 指向存放配置文件缓冲区的首地址
  DSPIoWrite(hDevice, Card_info, 4, 0x01980000, 4);//写'1'
DSPIoWrite(hDevice, Card_info, 4, 0x01980000, 0); //写'0'
Sleep(1);
DSPIoWrite(hDevice, Card_info, 4, 0x01980000, 4); //写'1'
//用于在nCONFIG脚产生逻辑低脉冲信号启动配置过程
  Sleep(1);
  for(DWORD i=0;i<Length;i++){//Length为配置文件长度
    data = *(databuf+i);     //取配置文件中的一个字节
    for(DWORD j=0;j<8;j++){//此循环写一个字节
      DSPIoWrite(hDevice, Card_info, 2, 0x02000000, data);
                          //写最低一位
      DSPIoRead(hDevice, Card_info, 2, 0x01980004, &resault);
      if ((resault & 0x0008)== 0) return (false);
      //检查nSTATUS状态，为低报错
      data = data>>1;
    }
  }
  for(DWORD i=0;i<32;i++){
    DSPIoWrite(hDevice, Card_info, 2, 0x02000000, 0);
//向DCLK提供32个时钟,确保芯片被正确初始化,进入用户模式
  }
return(true);
}
```

## 3 主机通过PCI接口对DSP软件动态加载

美国TI公司为DSP软件开发提供了集成环境CCS(Code Composer Studio)，用户可以用它编辑、编译程序，用连接器生成.out格式文件，再通过仿真器把此文件加载到目标板上，完成调试工作。然而在设备的应用现场，最终用户可能没有DSP仿真器，无法用JTAG接口实现系统的软件升级。TI公司的DSP有多种接口(如PCI)可以和主机连接，主机可以通过这些接口加载DSP软件。下面以TMS320C6205DSP为例介绍这一功能的实现。

通过CCS编译连接生成.out文件，实际上是COFF(Common Object File Format)格式文件，COFF文件结构参见表1。

**表1 COFF文件结构**

| |
|---|
| 文件头（File Header） |
| 可选头（Optional Header） |
| 段落头（Section Header）1 |
| …… |
| 段落头（Section Header）*n* |
| 段落数据（Section Data） |
| 重定位表（Relocation Directives） |
| 行号表（Line Numbers） |
| 符号表（Symbol Table） |
| 字符串表（String Table） |

其中，除了段落头可以有多个节(因为可以有多个段落)以外，其它的所有类型的节最多只能有一个。

如果直接将.out文件加载到内存中去，需要了解.out文件太多的细节，有一定的困难。TI公司提供了一个工具hex6x.exe，它可以将.out格式转换ASCII-HEX格式，命令为“hex6x xxx.out-a-o xxx.hex-memwidth 8”,这种格式相对简单，分析容易，把段地址小于0x10000段放在缓冲区progbuf，段地址大于0x10000的段放在缓冲区databuf，然后通过应用软件把这两个缓冲区的内容分别写到DSP的程序区和数据区，从而实现DSP软件的更改。实现上述这些功能的程序(已在C++ Builder 6.0下调试通过)如下：

```
extern "C" bool __export __pascal LoadProgram(HANDLE hDevice, CARD_INFO Card_info,void *hexfilebuf, DWORD hexfilelen)
{
  BYTE *sbuf, *tbuf;
  BYTE d0, d1, Addr[8], templen;
  DWORD Address,tbuf_count;
  DWORD offset, resault;
  BYTE *progbuf,*databuf;
  DWORD *pbuf, *dbuf;
  sbuf = (BYTE *)hexfilebuf;
                //指向存放HEX文件缓冲区的首地址
  progbuf = (BYTE *)GlobalAlloc(GMEM_FIXED, 0x10000);
                //分配64kB程序缓冲区
  databuf = (BYTE *)GlobalAlloc(GMEM_FIXED, 0x10000);
                //分配64kB数据缓冲区
  if((progbuf==NULL)||(databuf==NULL)){
    return(false);  //分配失败报错
                }
  memset(progbuf, 0, 0x10000);
  memset(databuf, 0, 0x10000);//两缓冲区写‘0’
  tbuf_count = 0;
  tbuf = (BYTE *)progbuf;
  for(DWORD i=0;i<hexfilelen;i++){//hexfilelen指HEX文件长度
    d0 = *(sbuf+i);
        if(((d0>=0x30)&&(d0<=0x39)) || ((d0>=0x41)&&(d0<=0x46)) || ((d0>=0x61)&&(d0<=0x66))){
  //检查d0是否为字符0-9、A-F或a-f
      i++;
      d1 = *(sbuf+i);
   *(tbuf+tbuf_count) = ASCIIToHex(d0)*0x10+ASCIIToHex(d1);
          //把字符d0和d1转换为一个十六进制数，存在tbuf中
      tbuf_count++; }
    if(d0==0x24){   //检查是否有新的段，标志为$字符
      i = i+2;
      for(DWORD j=0;j<8;i++,j++){//
        d0 = *(sbuf+i);
        if((d0<0x30)||((d0>0x39)&&(d0<0x41))||
  ((d0>0x46)&&(d0<0x61))||(d0>0x66)) {//检查段地址是否结束
          templen = j-1;
          break;
              }
        Addr[j] = ASCIIToHex(d0);// 转换为十六进制数
        templen = j;//段地址位数
          }
    Address = 0;
    for(BYTE j=0;j<=templen;j++){
      Address += Addr[j]*pow(0x10, templen-j);//计算段地址
          }
    if(Address>=0x10000)
      tbuf = (BYTE *)databuf;
            //段地址大于0x10000，段放在databuf中
    tbuf_count = Address&0x7FFFFFFF;
            //重置缓冲区的偏移地址
    }
}
```



© 1995-2005 Tsinghua Tongfang Optical Disc Co., Ltd. All rights reserved.

```
 ResetDSP(hDevice, Card_info); //复位DSP
 pbuf=(DWORD *)progbuf;
 dbuf=(DWORD *)databuf; //字节类型转为双字类型
 for(DWORD i=0;i<0x4000;i++){
                //把progbuf中的内容写到DSP的程序区
   offset = i*4;// DSP程序区的偏移
   resault = *(pbuf+i);//取数据
   DSPIoWrite(hDevice, Card_info, 4, offset, resault); //写数据
 }
 for(DWORD i=0;i<0x4000;i++){
                //把databuf中的内容写到DSP的数据区
   offset = 0x80000000+i*4; // DSP数据区的偏移
   resault = *(dbuf+i); //取数据
   DSPIoWrite(hDevice, Card_info, 4, offset, resault); //写数据
 }
 SendIntToDSP(hDevice, Card_info);//写完后给DSP发一个中断
 return(true);
}
```

## 4 结束语

本文主要介绍如何增强基于DSP系统的开放性设计，笔者在开发通用卫星接收处理设备时用了上面介绍的两种方法，利用主机通过DSP对FPGA的动态配置功能，实现了设备对几十种不同卫星线路卷积码的译码；利用主机通过PCI接口对DSP软件的动态加载功能，实现了系统软件的升级和设备功能的扩展，从而大大增强设备的通用性、灵活性和开放性。

## 参考文献


1 TMS320C6205 Fixed-point Digital Signal Processor . http://www-s.ti.com/sc/ds/tms320c6205.pdf, 2003

2 Configuring SRAM-based LUT Devices. http://www.altera.com/literature/an/an116.pdf ,2001

3 APEX 20K Programmable Logic Device Family Data Sheet .http://www.altera.com/literature/ds/ds_ap2.pdf ,2001

4 TMS320C6000 Assembly Language Tools User's Guide. http://focus.ti.com/lit/ug/spru186m/spru186m.pdf ,2003

5 李方慧，王　飞，何佩琨编著. TMS320C6000系列DSPs原理与应用.北京：电子工业出版社，2003


(上接第496页)

业务管理，以及系统监测等6大部分。各模块之间的联系见图3所示。

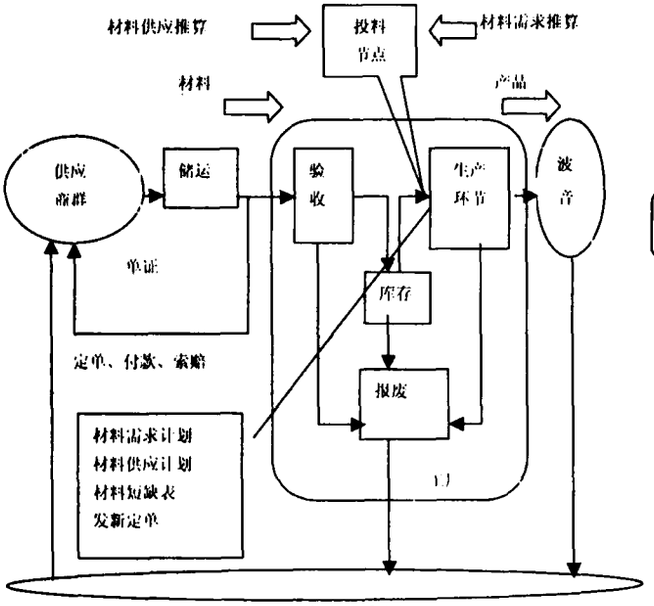


图2 系统物流示意图

本系统以人机会话的菜单方式，提供用户使用，系统在启动后，用户按屏幕各级菜单提示，点出需要处理的有关项目，使系统进入有关功能处理，直至完成。为了确保数据的安全，制定了网间数据交换体系，配置可单独评价的加密、数字签名、访问权限控制、数据完整性、业务流填充、路由控制、公证、鉴别审计等安全机制，在相应的网络层次和级别上设立密钥管理中心、访问控制中心、安全鉴别服务器、授权服务器等。系统通过实施：达到正确的材料定额(含正确的排料、套裁)；正确的按时配套(特别是标准件)控制；及时的库存信息发送，保证网上信息为最新有效；确定合理周转时间(物料及各制造过程都要细定)与月产量的关系，以及考虑有效的应急渠道和方法。系统设计的基本思想：对供料物流、资金流和信息流的全过程跟踪和管理。最终目标：从现在库存材料占用的流动资金减少30%。

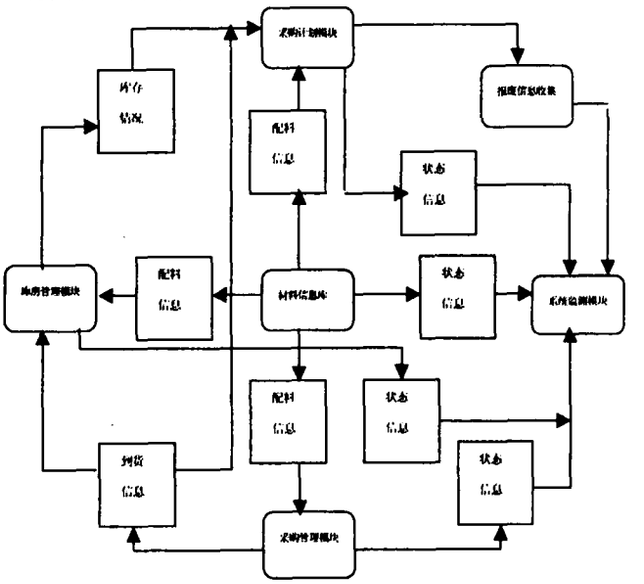

图3 主要功能模块之间的联系图

## 5 结束语

采购与库存管理系统经投入试运行，按照软件工程规范的要求，对系统进行了全面测试，各项性能均基本达到了系统设计的基本思想和用户提出的技术要求。实现了对飞机零部件采购与库存管理的全过程跟踪。通过这次物料采购与库存管理软件系统的开发与应用，既使上海飞机制造厂波音飞机水平尾翼项目上降低了物料采购成本，提高了实际生产效益，也为今后物流理念的进一步拓宽以及物流、资金流和信息流的日益融合，打下了坚实的基础。

## 参考文献


1 王 佐. 关于物流标准化问题的思考. 中国物流与采购. (208)

2 张正义. 现代企业物流的理念和技术发展趋势. 物流技术与应用. 2003-01

3 (日)藤木胜敏著.中国物流系统的未来. 物流技术与应用, 2003-07

4 (美)Waters D. 物流管理概论

5 张 铎. 现代物流信息系统建设




© 1995-2005 Tsinghua Tongfang Optical Disc Co., Ltd. All rights reserved.