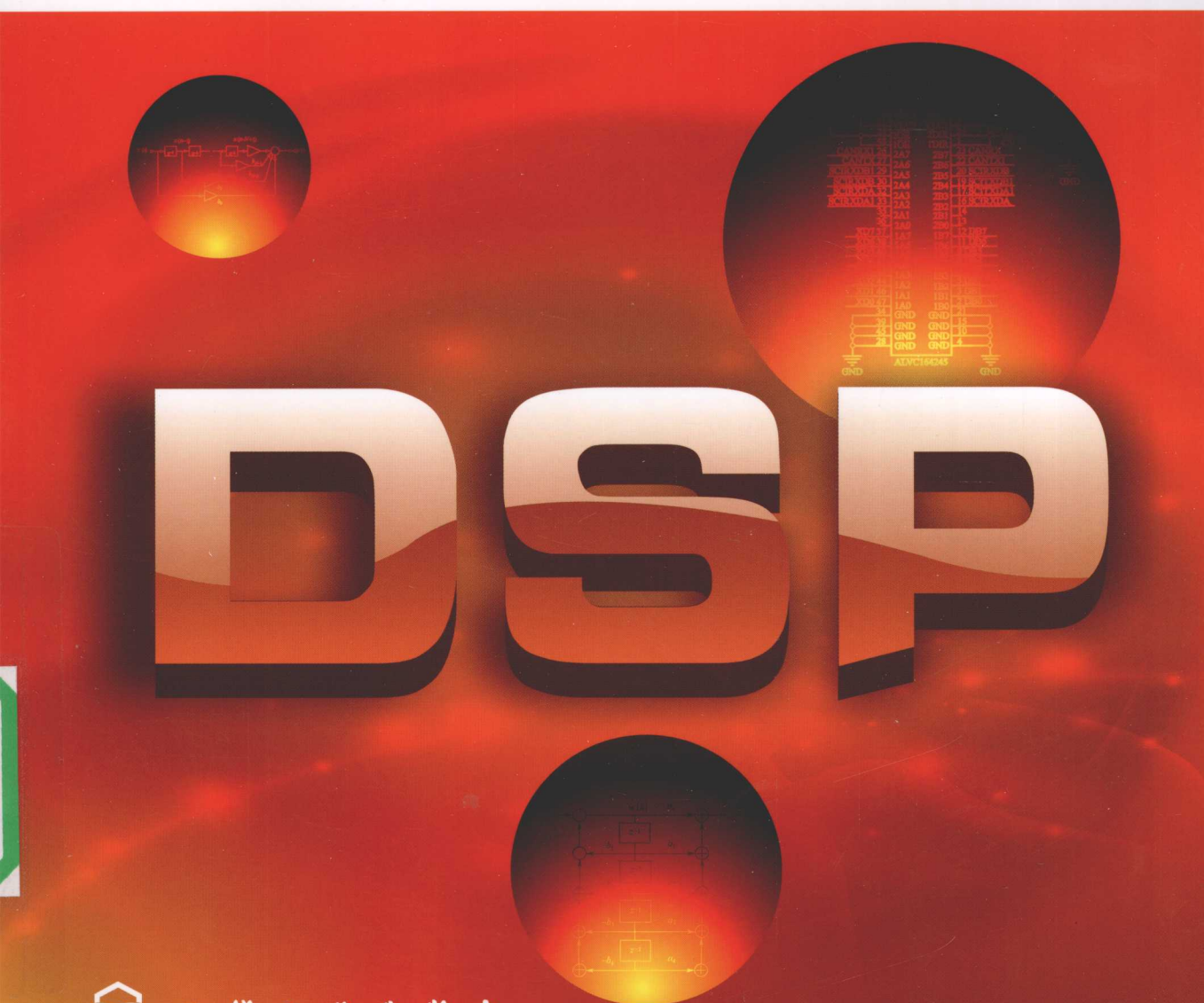
电子工程设计与应用百例系列

数字信号处理器
DSP应用100例

姜艳波　等编著

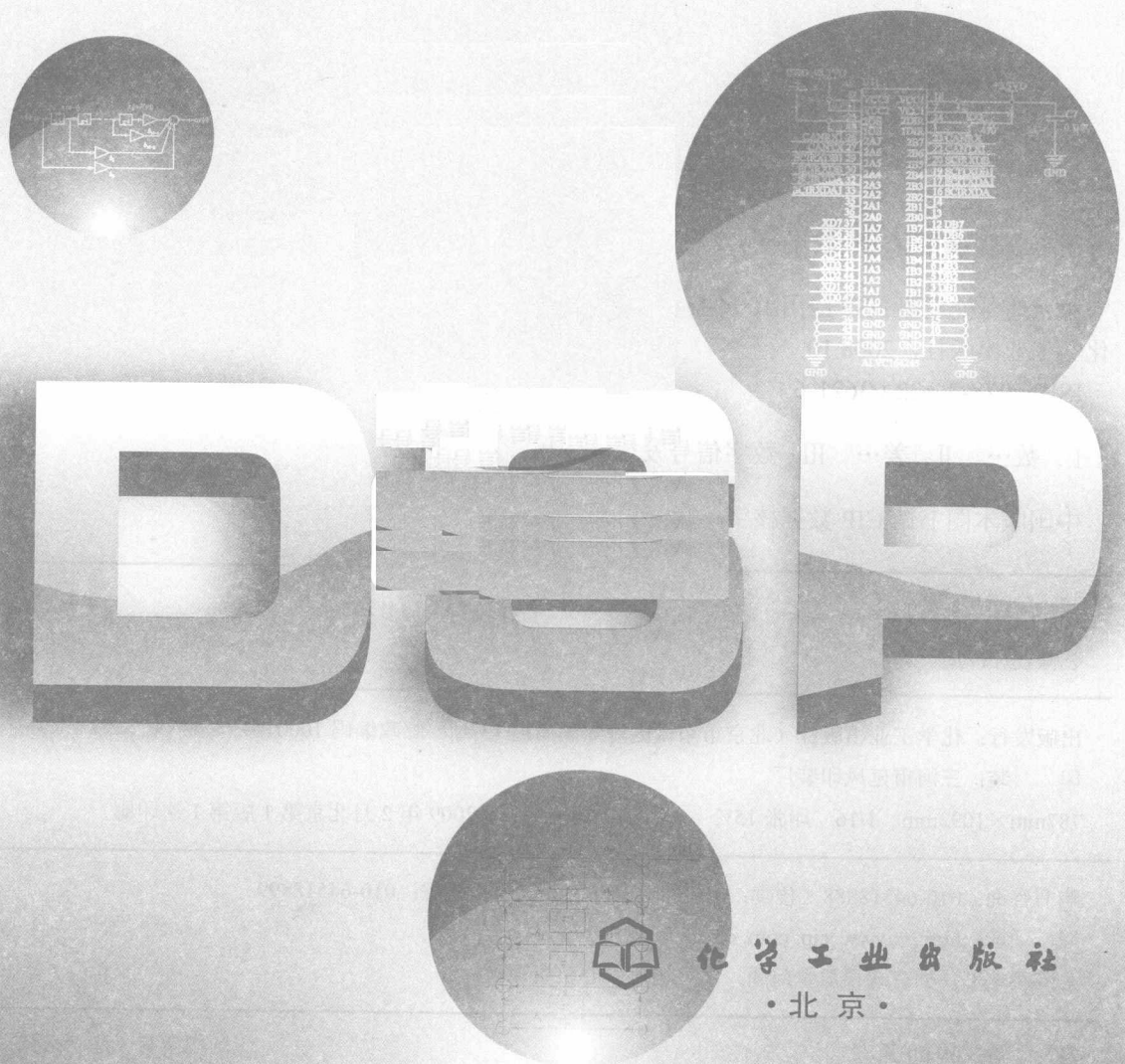


化学工业出版社

电子工程设计与应用百例系列

数字信号处理器DSP应用100例

姜艳波　等编著



化学工业出版社
·北京·

图书在版编目(CIP)数据

数字信号处理器DSP应用100例/姜艳波等编著. —北京：化学工业出版社，2008.10

ISBN 978-7-122-03621-6

Ⅰ. 数… Ⅱ. 姜… Ⅲ. 数字信号发生器 Ⅳ. TN911.72

中国版本图书馆CIP数据核字（2008）第133487号

责任编辑：刘 哲　　责任校对：周梦华

装帧设计：周 遥

出版发行：化学工业出版社（北京市东城区青年湖南街13号 邮政编码100011）

印　　装：三河市延风印装厂

787mm×1092mm 1/16 印张15½ 字数420千字 2009年2月北京第1版第1次印刷

购书咨询：010-64518888（传真：010-64519686） 售后服务：010-64518899

网　　址：http://www.cip.com.cn

凡购买本书，如有缺损质量问题，本社销售中心负责调换。

定　　价：39.00元

版权所有 违者必究

前言

数字信号处理器 DSP 是一门理论与实践结合的技术，在学习了 DSP 的结构体系与基本原理以后，必须通过配合一些典型的 DSP 实例，以加深对 DSP 软件、硬件的理解与掌握，同时学会 DSP 开发工具的使用，了解 DSP 应用系统的开发环境与开发过程，为继续学习 DSP 打下坚实的基础。

本书主要以 TI 公司的 TMS320LF2812 和 TMS320LF2407 两种 DSP 芯片为核心，通过对多个功能模块开发的实际过程的介绍，来讲解 DSP 开发的基本知识和开发实例。

TI 公司的 TMS320LF2812 和 TMS320LF2407 两种 DSP 芯片都是属于 TI 的 C2000 系列产品。在目前过程控制领域中，它是 TMS320 系列中的第二代产品。与传统的单片机相比，它具有功能强、资源丰富、功耗低等突出的性能；同时，这两种 DSP 芯片具有完美的性能及最佳的外设接口；它集成了闪存、高速 A/D 转换器、高性能的 CAN 模块等，具有较高的性价比，利用它可以降低开发难度，缩短面市时间，有效地降低了开发成本。

本书通过 100 个实例来详细地介绍 DSP 的各种工程实现和算法实现。

第 1 章 TI2000 系列 DSP 常用电路设计实例，主要介绍 DSP 的电源电路、复位电路、时钟电路、模数转换接口电路。第 2 章 CCS2000 使用和编程实例，通过 45 个实例来介绍 CCS2000 的开发及使用。第 3 章通用扩展语言（GEL）实例。GEL（General Extension Language）是一种类似于 C 语言的通用扩展语言，即是一种解释性语言，它可以创建 GEL 函数，以扩展 CCS 的用途。第 4 章 DSP 常用算法实例。算法是实现数字信号处理的重要手段，本章主要介绍 DSP 的常用算法。第 5 章与硬件结合程序实例，主要以综合实例的形式对 TI2000 系列 DSP 进行介绍。

本书内容丰富，讲解细致。实例中的代码全部编译通过，并配有相应的电路图。本书由姜艳波编写，参与编写的还有赵光、张玉平、李长林、兰婵丽、王波波、刘文涛、杨邵豫、张瑞雪、刘群等。

限于我们的水平，书中难免存在不妥之处，敬望读者批评指正！

编著者

目录

此为试读，需要完整PDF请访问：www.ertongbook.com

第1章

TI2000 系列 DSP 常用电路设计实例

一个独立工作的 DSP 芯片一般包括电源电路、复位电路、时钟电路、模数转换接口电路。本章将通过 12 个实例对 DSP 基本的硬件电路设计及应用进行详细的介绍。

【例 1-1】电源设计实例

1. 实例说明

本实例设计一个具有上电次序控制的电源电路。

电源是任何一个电气系统不可缺少的部分，DSP 应用电路系统一般为多电源系统。DSP 芯片内部的典型电源包括 CPU 内核电源、I/O 电源、PLL（phase locked loop）电源、Flash 编程电源、模拟电路电源五种电源，而后两种为 C2000 电源。

在电源设计的过程中，应该注意以下几点问题。

① 根据使用的芯片类型不同，内核电源、I/O 电源所需的电压也有一定的差距，这几种电源都要由各自的电源供电。

② 在进行电源设计时，模拟电路和数字电路部分要独立供电，数字地与模拟地分开，遵循“单点”接地的原则。

③ DSP 供电电源在设计时，内核电源与端口电源的电压是不同的，它们需要两种供电电源，这时就必须要考虑它们之间的配合问题。

2. 电路应用

在上电过程中，可能会遇到以下两种情况：一是内核先获得供电，外围没有得到供电，这时就会没有输入输出，但对芯片不会产生损坏；二是外围 I/O 接口先得到供电，内核后得到供电，则有可能会导致 DSP 和外围引脚同时作为输出端，如果此时双方输出的值是相反的，那么两输出端会因反向驱动可能出现大电流，从而影响器件的寿命，甚至损坏器件。

在掉电时也该注意，如果内核先掉电，出现的电流可能比较大，因此一般要求 CPU 内核电源先于 I/O 电源上电，后于 I/O 电源掉电。但是 CPU 内核电源与 I/O 电源供电时间相差不能过长（一般应小于 1 s，否则会影响器件的寿命或损坏器件）。在 CPU 内核电源与 I/O 电源之间还加了一个肖特基二极管，它的主要作用是保护 DSP 器件。

具有上电次序控制的 DSP 电源电路如图 1-1 所示。

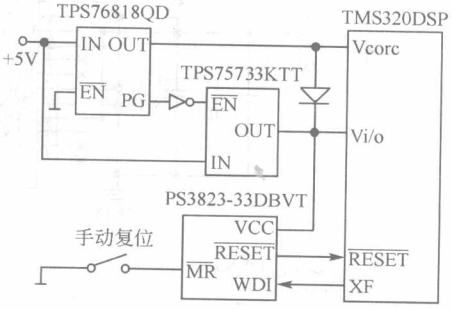


图 1-1　电源电路

【例 1-2】供电电路设计实例

1. 实例说明

本实例应用北京三恒星科技公司的 DSP2407 开发板设计供电电路。

2. 电路应用

电源由核心板引入，电源插孔 J2 标识为内正外负，+5V 稳压直流电源输入，LD1117-3.3 电源转换芯片作为 5V 转 3.3V 的高性能稳压芯片。

具体电路图如图 1-2 所示。

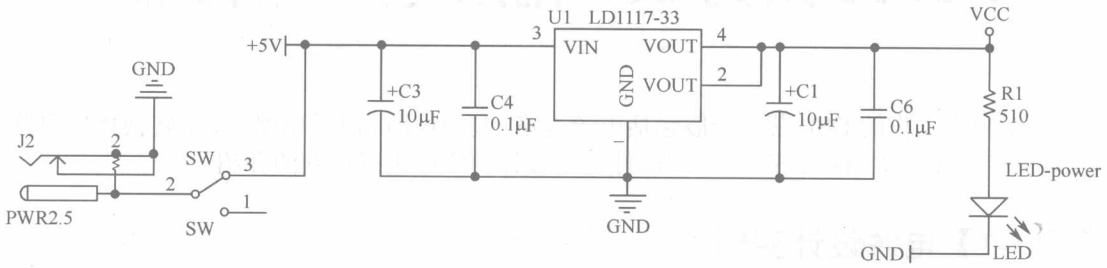


图 1-2　供电电路

【例 1-3】电源与滤波电路设计实例

1. 实例说明

本实例应用北京三恒星科技公司的 DSP2812 开发板设计电源与滤波电路。

2. 电路应用

本实例采用外部 5V 直流电压供电，电源插孔标识为内正外负。电源芯片 TPS767D318 为双电源输出，一路为 3.3V、一路为 1.8V，分别为外围电路和 CPU 供电，每路电源的最大输出电流为 1A。

具体电路如图 1-3 所示。

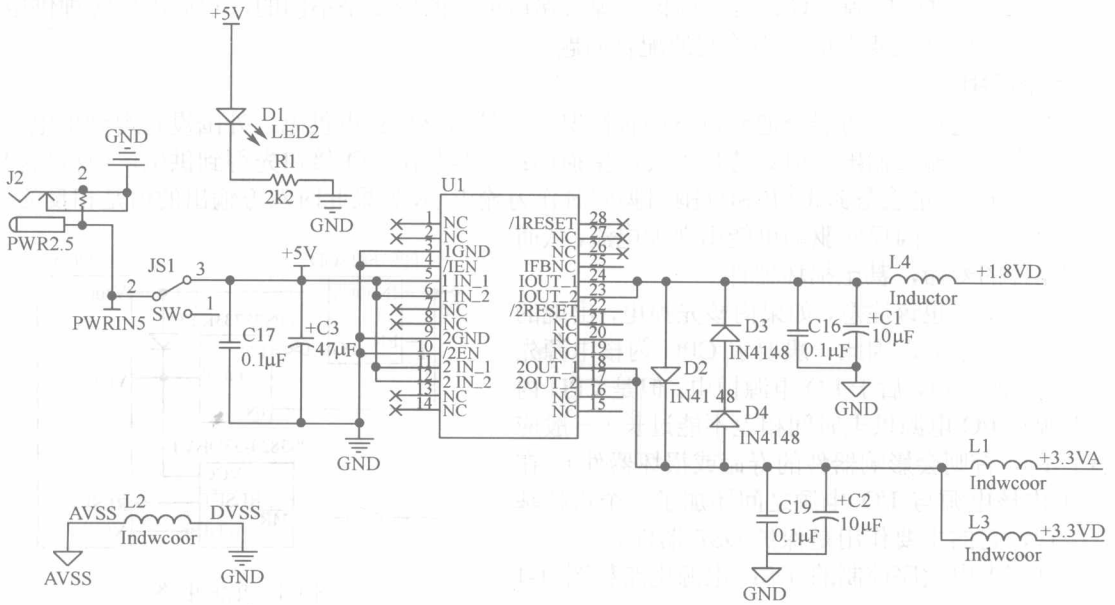


图 1-3　电源与滤波电路

【例 1-4】缓冲与电平转换设计实例

1. 实例说明

本实例应用北京三恒星科技公司的 DSP2812 开发板设计缓冲与电平转换电路。

2. 电路应用

本实例选用 TI 公司的 ALVC164245 电平转换器，双电压（一边是 3.3 V，另一边是 5 V）供电的双向驱动器，实现电平转换。在嵌入式系统设计中经常碰到 3.3V 和 5V 电路混合的问题，现在很多处理器都是 3.3V 的，必须使用该总线转换器件，驱动 5V 电路芯片。

具体电路图如图 1-4 所示。

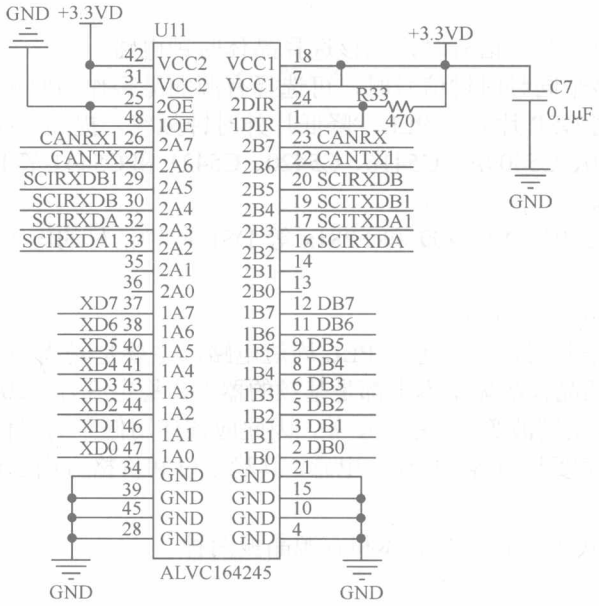


图 1-4　缓冲与电平转换电路

【例 1-5】复位电路设计实例

1. 实例说明

本实例设计一个 RC 复位电路。该复位电路的上电复位一般在芯片的 RESET 引脚上置 100～200ms 的低电平脉冲。

在系统中加入电源监控和复位电路主要有以下几点作用。

① 为了保证 DSP 芯片在电源未达到要求的电平时，不会产生不受控制的状态。

② 由该复位电路确保在系统加电的过程中，在内核电压和外围端口电压达到要求之前，DSP 芯片始终处于复位状态，直到内核电压和外围接口电压达到所要求的电平。

③ 如果电源电压降到门限值以下，则强制芯片进入复位状态，确保系统稳定工作。

在设计复位电路时，一方面应确保复位低电平时间足够长（一般需要 20ms 以上），以保证 DSP 可靠复位；另一方面应保证复位电路的稳定性良好，防止 DSP 误复位。主要是应保证复位输入端（RS）低电平至少持续 6 个时钟周期，即若时钟为 20MHz 时，低电平持续的时间为 300ns。

2. 电路应用

在上电后，系统的晶振需要几百毫秒的稳定期，所以一般设为 100～300ms。设计时可将上电复位和手动复位两个信号经过逻辑相与，后送到 DSP 的复位输入引脚。具有上电延迟复位和手动复位功能的复位电路如图 1-5 所示。系统调试和系统运行出现故障时可以方便地使用手动复位。

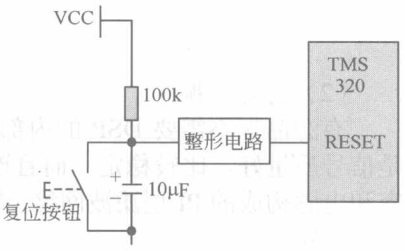


图 1-5　复位电路图

【例 1-6】时钟电路设计实例

1. 实例说明

时钟电路的设计是 DSP 系统设计中的重要组成部分。

时钟电路设计遵守以下几点原则。

① 当系统要求多个不同频率的时钟信号时，首先应选可编程时钟芯片，这样有利于时钟信号的同步。

② 当系统要求单一时钟信号时，应该选择晶体时钟电路。

③ 当系统要求多个同频时钟信号时，可选择有源的晶振作为时钟电路。

④ 应该尽量使用 DSP 片内的 PLL，降低片外时钟频率，提高系统的稳定性。

⑤ C6000、C5510、C5409A、C5416、C5420、C5421 和 C5441 等 DSP 片内无振荡电路，不能用晶体时钟电路。

⑥ VC5401、VC5402、VC5409 和 F281x 等 DSP 时钟信号的电平为 1.8V，建议采用晶体时钟电路。

时钟电路设计的几点注意事项。

① 当 DSP 需要倍频时，要配置好 PLL 周边电路，主要是隔离和滤波。

② 20MHz 以下的晶体晶振基本上都是基频的器件，稳定度好，20MHz 以上的大多是谐波的（如 3 次谐波、5 次谐波等），稳定度差，因此应该使用低频的器件，因为倍频用的 PLL 电路需要的周边配置主要是电容、电阻、电感，其稳定度和价格方面远远好于晶体晶振器件。

2. 电路应用

时钟电路设计方式主要有无源晶体和有源晶振两种。

（1）无源晶体

无源晶体需要用 DSP 片内的振荡器。无源晶体无需考虑电压的问题，信号电平是可变的，是由起振电路来决定的，同样的晶体可以适用于多种电压，可用于多种不同时钟信号电压要求的 DSP，而且价格通常也较低，因此对于一般的设计建议使用无源晶体。无源晶体与晶振相比其缺点主要是信号质量较差，通常需要精确匹配外围电路（用于信号匹配的电容、电感、电阻等），更换不同频率的晶体时周边配置电路需要做相应的调整。建议采用精度较高的石英晶体，尽可能不要采用精度低的陶瓷晶体。

本例采用内部振荡器方式，由一个 20MHz 的石英晶振提供参考频率，用两个电容滤除波。电路图如图 1-6 所示。

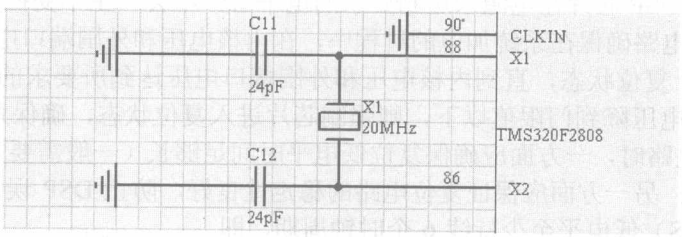


图 1-6　内部振荡器电路

（2）有源晶振

有源晶振不需要 DSP 的内部振荡器，将外部时钟源直接输入 DSP 芯片内部。它的优点是信号质量好，比较稳定，而且连接方式相对简单（主要是做好电源滤波，通常使用一个电容和电感构成的 PI 型滤波网络，输出端用一个小阻值的电阻过滤信号即可），不需要复杂的配置电路。

有源晶振通常的用法：1 脚悬空，2 脚接地，3 脚接输出，4 脚接电压。与无源晶体相比，

此为试读，需要完整PDF请访问：www.ertongbook.com

有源晶振的缺点是信号电平是固定的，需要选择合适的输出电平，灵活性较差，而且价格相对较高。对于时序要求敏感的应用，还是应用有源的晶振比较好，这样就可以选用比较精密的晶振，甚至是高档的温度补偿晶振。有些 DSP 内部没有起振电路，只能使用有源的晶振，如 TI 的 6000 系列等。有源晶振相比于无源晶体通常体积较大，但现在许多有源晶振是表贴的，体积和晶体相当，有的甚至比许多晶体还要小。本例采用封闭好的晶体振荡器，这种方法使用很方便，得到了广泛的应用。

具体设计方法是在 4 脚加上 5V 电压，2 脚接地，1 脚悬空，就可在 3 脚得到所需的时钟。电路图如图 1-7 所示，该电路图为顶视图。

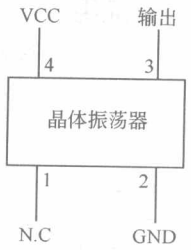


图 1-7　晶体振荡电路

【例 1-7】2407 时钟电路设计实例

1. 实例说明

本实例应用北京三恒星科技公司的 DSP2407 开发板设计时钟电路。

2. 电路应用

该电路采用的是内部振荡器方式，选用的外部晶振为 20MHz 的电路。如果外部晶振大小改变，滤波器 PLLF 和 PLLF2 之间电阻电容的参数也必须改变。下面列出几个常用的参数值，如表 1-1 所示。

表 1-1　常用电容参数值

XTAL1/CLKIN 频率/MHz	R43/Ω（±5%误差）	C336/μF（±20%误差）	C347/μF（±20%误差）
8	9.1	0.022	1
10	11	0.015	0.68
12	13	0.01	0.47
15	16	0.0068	0.33
16	18	0.0056	0.27
20	24	0.0033	0.15

在本例中，为增强电路的抗干扰能力，建议使用 8～16MHz 的晶振，然后内部倍频。具体电路图如图 1-8 所示。

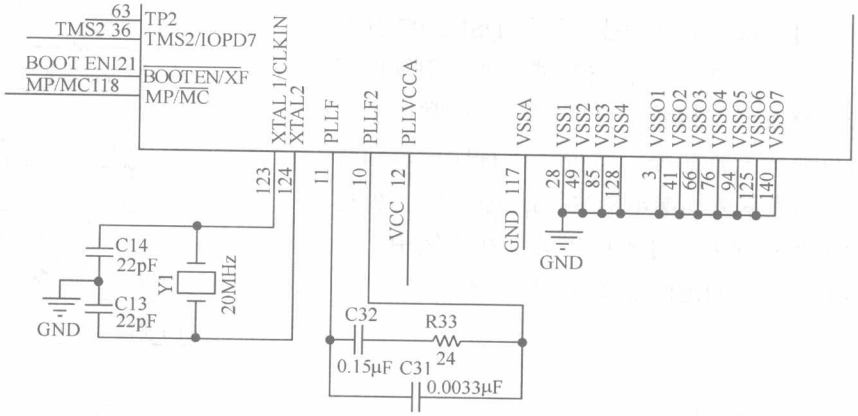


图 1-8　时钟振荡电路

【例 1-8】JTAG 仿真口电路设计实例

1. 实例说明

JTAG（Joint Test Action Group，IEEE1149．1）称为连接测试组接口。JTAG 接口用于

连接最小系统板和仿真器，实现对 DSP 的访问。

2. 电路应用

JTAG 口连接需要和仿真器上给出的引脚一致。TI 公司仿真器的 14 脚 JTAG 口的引脚如图 1-9 所示。

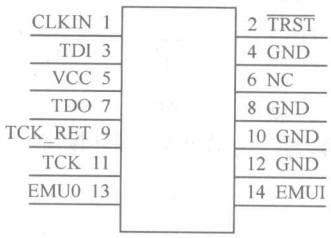


图 1-9　14 脚 JTAG 口引脚图

通常情况，最小系统板引出的双排 14 个引脚应与图 1-9 所示的引脚结构一致。图 1-9 的引脚间隔为 0.1in，引脚宽度为 0.025in，引脚长度为 0.235in。

根据开发板和仿真器之间连接电缆的距离，可以采用不同的连接方法。

① 如果开发板和仿真器之间的连接电缆不超过 6in，电路连接方法如图 1-10 所示。在连接电路的过程中，DSP 的 EMU0 和 EMU1 引脚需要上拉电阻，电阻阻值一般为 4.7kΩ 和 10kΩ。

② 如果 DSP 和仿真器之间的连接电缆超过 6in，应采用图 1-11 所示的电路连接方法。

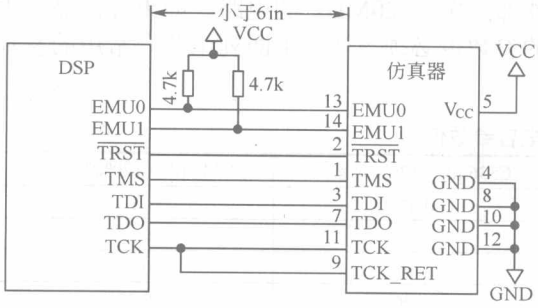


图 1-10　小于 6in 的 JTAG 连接方法

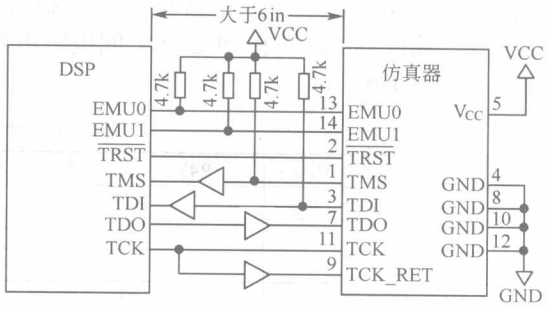


图 1-11　大于 6in 的 JTAG 连接方法

【例 1-9】定时器 LED 设计实例

1. 实例说明

本实例应用北京三恒星科技公司的 DSP2407 开发板实现循环点亮 8 个发光二极管功能，通过用软件延时的方法来调整发光二极管的延时间隔。在掌握了本程序后，可以充分发挥想象，改变一下程序内容和灯的接法（比如用光耦驱动继电器来控制霓虹灯），就可做出各种变化非凡的大型灯光广告牌。在本例中，灯的闪烁频率由软件中的定时器 Timer1 设置。

2. 电路应用

（1）硬件电路

具体的硬件电路如图 1-12 所示。

（2）软件设计

通过软件设计，设置定时器 Timer1 来控制 LED 的闪烁。

主要程序如下：

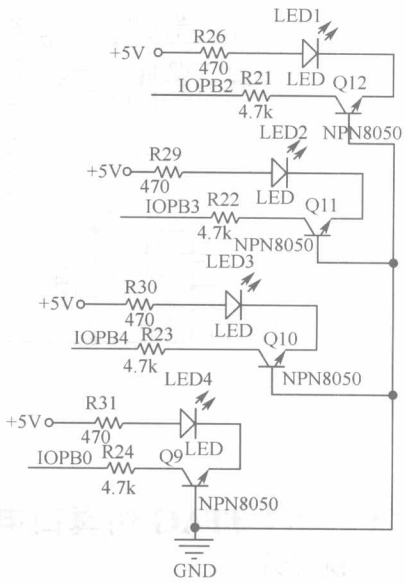


图 1-12　LED 闪烁电路图

```
;/*********************************************/
;/*  文件名称: LF2407A.H                      */
;/*********************************************/

;/*********************************************/
;/* CPU 内核寄存器                            */
;/*********************************************/
_IMR        .set    00004H          ;中断屏蔽寄存器
_IFR        .set    00006H          ;中断标志寄存器

;/*********************************************/
;/*  系统配置和中断寄存器                     */
;/*********************************************/
_SCSR1      .set    07018H          ;系统控制/状态寄存器 1
_SCSR2      .set    07019H          ;系统控制/状态寄存器 2

_PIVR       .set    0701EH          ;外部中断向量寄存器
;/*********************************************/
;/*  外部中断配置寄存器                       */
;/*********************************************/
_XINT1CR    .set    07070H          ;外部中断 1 控制寄存器
_XINT2CR    .set    07071H          ;外部中断 2 控制寄存器

;/*********************************************/
;/*  数字 I/O 寄存器                          */
;/*********************************************/
_MCRA       .set    07090H          ;PA 口的控制寄存器
_MCRB       .set    07092H          ;PB 口的控制寄存器
_MCRC       .set    07094H          ;PE 口和 PF 口的控制寄存器
_PADATDIR   .set    07098H          ;PA 口的方向数据寄存器
_PBDATDIR   .set    0709AH          ;PB 口的方向数据寄存器
_PCDATDIR   .set    0709CH          ;PC 口的方向数据寄存器
_PEDATDIR   .set    07095H          ;PE 口的方向数据寄存器
_PFDATDIR   .set    07096h          ;I/O 口的方向和数据寄存器
;/*********************************************/
;/*  看门狗寄存器                             */
;/*********************************************/
_WDCNTR     .set    07023H          ;WD 计数器
_WDKEY      .set    07025H          ;WD 键值
_WDCR       .set    07029H          ;WD 控制寄存器

;/*********************************************/
;/* SCI 寄存器                                */
;/*********************************************/
_SCICCR     .set    07050H          ;SCI 通信控制寄存器
_SCICTL1    .set    07051H          ;SCI 控制寄存器 1
_SCIHBAUD   .set    07052H          ;SCI 波特率寄存器（高位）
_SCILBAUD   .set    07053H          ;SCI 波特率寄存器（低位）
_SCICTL2    .set    07054H          ;SCI 控制寄存器 2
_SCIRXST    .set    07055H          ;SCI 接收状态寄存器
_SCIRXEMU   .set    07056H          ;SCI 仿真数据缓冲寄存器
_SCIRXBUF   .set    07057H          ;SCI 接收数据缓冲寄存器
```

```
_SCITXBUF       .set    07059H      ;SCI 发送数据缓冲寄存器
_SCIPRI         .set    0705FH      ;SCI 中断优先级控制寄存器

;/*********************************************/
;/*ADC 寄存器                                  */
;/*********************************************/
_ADCTRL1        .set    070A0H      ;ADC 控制寄存器 1
_ADCTRL2        .set    070A1H      ;ADC 控制寄存器 2
_MAXCONV        .set    070A2H      ;最大转换通道寄存器
_CHSELSEQ1      .set    070A3H      ;通道选择时序控制寄存器 1
_CHSELSEQ2      .set    070A4H      ;通道选择时序控制寄存器 2
_CHSELSEQ3      .set    070A5H      ;通道选择时序控制寄存器 3
_CHSELSEQ4      .set    070A6H      ;通道选择时序控制寄存器 4
_AUTO_SEQ_SR    .set    070A7H      ;自动时序状态寄存器
_RESULT0        .set    070A8H      ;转换结果寄存器 0
_RESULT1        .set    070A9H      ;转换结果寄存器 1
_RESULT2        .set    070AAH      ;转换结果寄存器 2
_RESULT3        .set    070ABH      ;转换结果寄存器 3
_RESULT4        .set    070ACH      ;转换结果寄存器 4
_RESULT8        .set    070B0H      ;转换结果寄存器 8
_RESULT9        .set    070B1H      ;转换结果寄存器 9

;/*********************************************/
;/*    CAN 寄存器                               */
;/*********************************************/
_CANMDER        .set    07100H      ;缓冲使能/通信方向
_CANTCR         .set    07101H      ;发送控制
_CANRCR         .set    07102H      ;接收控制
_CANMCR         .set    07103H      ;主控
_CANBCR2        .set    07104H      ;位配置 2
_CANBCR1        .set    07105H      ;位配置 1
_CANESR         .set    07106H      ;错误状态
_CANGSR         .set    07107H      ;全局状态
_CANCEC         .set    07108H      ;发送/接收错误计数
_CANIFR         .set    07109H      ;中断标志
_CANIMR         .set    0710AH      ;中断屏蔽
_CANLAM0H       .set    0710BH      ;接收屏蔽:mbox0/1
_CANLAM0L       .set    0710CH      ;接收屏蔽:mbox0/1
_CANLAM1H       .set    0710DH      ;接收屏蔽:mbox2/3
_CANLAM1L       .set    0710EH      ;接收屏蔽:mbox2/3

DP_CANBOX       .set    0E4H
_CANID0L        .set    07200H      ;mbox 0 的 ID(低位)
_CANID0H        .set    07201H      ;mbox 0 的 ID(高位)
_CANCTRL0       .set    07202H      ;RTR 和 DLC
_CANBX0A        .set    07204H
_CANBX0B        .set    07205H
_CANBX0C        .set    07206H
_CANBX0D        .set    07207H

_CANID1L        .set    07208H      ;mbox 1 的 ID(低位)
_CANID1H        .set    07209H      ;mbox 1 的 ID(高位)
```

```
_CANCTRL1       .set    0720AH      ;RTR 和 DLC
_CANBX1A        .set    0720CH
_CANBX1B        .set    0720DH
_CANBX1C        .set    0720EH
_CANBX1D        .set    0720FH

_CANID2L        .set    07210H      ;mbox 2 的 ID(低位)
_CANID2H        .set    07211H      ;mbox 2 的 ID(高位)
_CANCTRL2       .set    07212H      ;RTR 和 DLC
_CANBX2A        .set    07214H
_CANBX2B        .set    07215H
_CANBX2C        .set    07216H
_CANBX2D        .set    07217H

_CANID3L        .set    07218H      ;mbox 3 的 ID(低位)
_CANID3H        .set    07219H      ;mbox 3 的 ID(高位)
_CANCTRL3       .set    0721AH      ;RTR 和 DLC
_CANBX3A        .set    0721CH
_CANBX3B        .set    0721DH
_CANBX3C        .set    0721EH
_CANBX3D        .set    0721FH

_CANID4L        .set    07220H      ;mbox 4 的 ID(低位)
_CANID4H        .set    07221H      ;mbox 4 的 ID(高位)
_CANCTRL4       .set    07222H      ;RTR 和 DLC
_CANBX4A        .set    07224H
_CANBX4B        .set    07225H
_CANBX4C        .set    07226H
_CANBX4D        .set    07227H

_CANID5L        .set    07228H      ;mbox 5 的 ID(低位)
_CANID5H        .set    07229H      ;mbox 5 的 ID(高位)
_CANCTRL5       .set    0722AH      ;RTR 和 DLC
_CANBX5A        .set    0722CH
_CANBX5B        .set    0722DH
_CANBX5C        .set    0722EH
_CANBX5D        .set    0722FH

;/*********************************************/
;/*    EVA 控制寄存器                          */
;/*********************************************/
_GPTCONA        .set    07400H      ;全局通用定时器控制寄存器
_T1CNT          .set    07401H      ;T1 计数寄存器
_T1PR           .set    07403H      ;通用定时器的周期寄存器
_T1CON          .set    07404H      ;定时器控制寄存器
_T2CNT          .set    07405H      ;T2 计数寄存器
_T2PR           .set    07407H      ;T2 的周期寄存器
_T2CON          .set    07408H      ;T2 的控制寄存器
_ACTRA          .set    07413H
_DBTCONA        .set    07415H
_CMPR1          .set    07417H
_COMCONA        .set    07411H
```

```
_EVAIMRA        .set    0742CH
_EVAIMRB        .set    0742DH
_EVAIFRA        .set    0742FH
_EVAIFRB        .set    07430H

_GPTCONB        .set    07500H
_T3CNT          .set    07501H
_T3CMPR         .set    07502H
_T3PR           .set    07503H
_T3CON          .set    07504H

_EVBIMRA        .set    0752CH
_EVBIFRA        .set    0752FH

;/*********************************************/
;/*    I/O 空间映射寄存器                     */
;/*********************************************/
_WSGR           .set    0FFFFH      ;等待状态发生器控制寄存器
/*Main.c,LED 与定时器程序*/
#include "global.c"
void SystemInit();
void Timer1Init();
void KickDog();
int numled=200;

main()
{

    SystemInit();                           //系统初始化

    MCRA=MCRA & 0xC0FF;                     //IOPB0~6 设为 I/O 口模式
    PBDATDIR=0xFFC2;                        //所有 LED=0
    PBDATDIR=PBDATDIR |0x003D;              //所有 LED=1
    Timer1Init();                           //定时器初始化
    asm(" CLRC INTM ");
    while(1);

}

void SystemInit()
{

      asm(" SETC    INTM ");                //关闭总中断
      asm(" CLRC   SXM   ");                //禁止符号位扩展
      asm(" CLRC   CNF   ");                //B0 块映射为片上 DARAM
      asm(" CLRC   OVM   ");                //累加器结果正常溢出
      SCSR1=0x83FE;                         //系统时钟 CLKOUT=20×2=40M
                                            //打开 ADC,EVA,EVB,CAN 和 SCI 的时钟,系统时钟
                                            //CLKOUT=40M
      WDCR=0x006F;                          //禁止看门狗,看门狗时钟 64 分频
      KickDog();                            //初始化看门狗
     IFR=0xFFFF;                            //清除中断标志
```


此为试读,需要完整PDF请访问: www.ertongbook.com