西安科技大学
XI'AN UNIVERSITY OF SCIENCE AND TECHNOLOGY

ARM 课程设计

设计名称：基于 LPC2131 的频率计

所在学院：电气与控制工程学院

专业班级：测控技术与仪器 0902

学生姓名：雷军　　魏璐

学生学号：0906070225　　0906070227

指导老师：黄梦涛　　李红岩

完成时间：2013/01/04

目录

原创力文档
max.book118.com
预览与源文档一致,下载高清无水印

基于 LPC2131 的频率计

一、概述

1.1 课题设计的背景

数字频率计[1](DFM)是电子测量与仪表技术最基础的电子仪表类别之一， 数字频率计是计算机、通讯设备、音频视频等科研生产领域不可缺少的测量仪器，而且它是数字电压表(DVM)必不可少的部件。当今数字频率计不仅是作为电压表、计算机、天线电广播通讯设备、工艺过程自动化装置。多种仪表仪器与家庭电器等许多电子产品中的数据信息输出显示器反映到人们眼帘。集成数字频率计由于所用元件少、投资少,体积小,功耗低,且可靠性高,功能强,易于设计和研发,使得它具有技术上的实用性和应用的广泛性。不论从我们用的彩色电视机、电冰箱，DVD，还有我们现在家庭常用到的数字电压表数字万用表等等都包含有频率计。现在频率计已是向数字智能方向发展，即可以很精确的读数也精巧易于控制。数字频率计已是现在频率计发展的方向,它不仅可以很方便的读数,而且还可以使频率的测量范围和测量准确度上都比模拟先进.而且频率计的使用已是很多的方面,数字卫星、数字通讯等高科技的领域都有应用，今天数字频率计的发展已经不仅仅是一个小电子产品的发展也是整个民族乃至整个国家的发展，所以频率计的发展是一个整体的趋势。

而从民族产业上来说，我们在这种产业中还落后于西方发达国家，这将会关系到民族产业的兴衰。所以我们必须很重视当前的情况，学习发达国家的先进技术以发展本国的产业。

1.2 课题研究的目的和意义

数字频率计[1]是计算机、通讯设备、音频视频等科研生产领域不可缺少的测量仪器。随着人们文化生活水平的提高，加上现在中国国力的上升，人民在不断的追求高质量生活的同时大都在密切的关注着我们的民族产业的发展前景。而频率计的发展虽是一个极小部分但也可以反映出我国民族产业发展的现状。

我国在很多的方面都已不在是过去那个很贫穷落后的国家，但是关系着我们国计民生的民族产业的发展却是不尽人意，不能不成为今天令人注目的焦点。

通过本次课程设计，运用已学的课程知识，根据题目要求进行软硬件系统的设计和调试，对《ARM 嵌入式系统基础教程》[6]课程中涉及的芯片结构、控制原理、硬件和编程方面有一定的感性认识和实践操作能力，从而加深对本课程知识点的理解，使自身应用只是能力、设计能力、调试能力以及报告撰写能力等方面有显著的提高。

此次课程设计是用 ARM 的定时器/计数器的定时和计数功能，外部扩展 6 位 LED 数码管，求累计每秒进入 ARM 的外部脉冲个数，用 LED 数码管显示出来。或用上位机显示。

1.3、ARM 开发板(简介)

ARM 公司是专门从事基于 RISC 技术芯片设计开发的公司，作为知识产权供应商，本身不直接从事芯片生产，靠转让设计许可由合作公司生产各具特色的芯片，世界各大半导体生产商从 ARM 公司购买其设计的 ARM 微处理器核，根据各自不同的应用领域，加入适当的外围电路，从而形成自己的 ARM 微处理器芯片进入市场。目前，全世界有几十家大的半导体公司都使用 ARM 公司的授权，因此既使得 ARM 技术获得更多的第三方工具、制造、软件的支持，又使整个系统成本降低，使产品更容易进入市场被消费者所接受，更具有竞争力。

ARM 架构包含了以下精简指令集处理器的特性：

- 读取 / 储存 架构
- 不支援地址不对齐内存存取（ARMv6 内核现已支援）
- 正交指令集（任意存取指令可以任意的寻址方式存取数据 Orthogonal instruction set）
- 大量的 16 × 32-bit 寄存器阵列（register file）
- 固定的 32 bits 操作码（opcode）长度，降低编码数量所产生的耗费，减轻解码和流水线化的负担。
- 大多均为一个 CPU 周期执行。

为了补强这种简单的设计方式，相较于同时期的处理器如 Intel 80286 和 Motorola 68020，还多加了一些特殊设计：

- 大部分指令可以条件式地执行，降低在分支时产生的负重，弥补分支预测器（branch predictor）的不足。
- 算数指令只会在要求时更改条件编码（condition code）
- 32-bit 筒型位移器（barrel shifter）可用来执行大部分的算数指令和寻址计算而不会损失效能
- 强大的索引寻址模式（addressing mode）
- 精简但快速的双优先级中断子系统，具有可切换的暂存器组

1.4、LPC2131 微控制器

1、简介

LPC2131/2132/2138 是基于一个支持实时仿真和跟踪的 16/32 位 ARM7TDMI-STM CPU，并带有 32kB、64kB 和 512kB 嵌入的高速 Flash 存储器。128 位宽度的存储器接口和独特的加速结构使 32 位代码能够在最大时钟速率下运行。对代码规模有严格控制的应用可使用 16 位 Thumb 模式将代码规模降低超过 30%，而性能的损失却很小。较小的封装和很低的功耗使 LPC2131/2132/2138 特别适用于访问控制和 POS 机等小型应用中；由于内置了宽范围的串行通信接口和 8/16/32kB 的片内 SRAM，它们也非常适合于通信网关、协议转换器、软件

modem、语音识别、低端成像，为这些应用提供大规模的缓冲区和强大的处理功能。多个 32 位定时器、1 个或 2 个 10 位 8 路的 ADC、10 位 DAC、PWM 通道、47 个 GPIO 以及多达 9 个边沿或电平触发的外部中断使它们特别适用于工业控制应用以及医疗系统。

2、主要性能

a、 8/16/32kB 的片内静态 RAM 和 32/64/512kB 的片内 Flash 程序存储器。128 位宽度接口/加速器可实现高达 60 MHz 工作频率。

b、 1 个（LPC2131/2132）或 2 个（LPC2138）8 路 10 位的 A/D 转换器，共提供 16 路模拟输入，每个通道的转换时间低至 2.44us。

原创力文档
max.book118.com
预览下载高清无水印

c、 1 个 10 位的 D/A 转换器，可产生不同的模拟输出。（仅适用于 LPC2132/2138）

d、 2 个 32 位定时器/计数器（带 4 路捕获和 4 路比较通道）、PWM 单元（6 路输出）和看门狗。

e、多个串行接口，包括 2 个 16C550 工业标准 UART、2 个高速 I2C 接口（400 kbit/s）、SPITM 和具有

二、 方案设计

方案 1：通过板内的 1 个定时器，完成发射频率与频率计数。板内自带的定时器向另一个定时器发送方波频率，由另一个定时器接受并进行计数。外接液晶屏 1602 显示频率，并且显示的程序可由板内的 2 个按键切换或进行中断。

方案 2：由外部频率发射器传送出频率后经 LPC2131 定时器对频率进行计数。然后由液晶屏 1602 对频率进行计数显示。最后利用开发板内自带的按键控制频率并对频率进行中断或刷新。

因为方案 1 节省资源，并且板内自带的定时器可以提供更方便快捷的程序更改频率，调试也可只对用程序进行修改，方案 1 的按键功能也可方便我们进行程序扩展。综上所述，我们决定使用方案 1。方案 1 报告可以节省资源外，编译的程序上也带给我们一定有利之处，例如在程序出错的时候我们判断错误的来源也可以从少量的器件中一一排除，方案 2 外接的频率发射机不仅要昂贵的经济消耗以外，所带给我们的程序编译上也没有较好快捷的方法进行修改。

三、 硬件设计

3.1 硬件设计原理图及其介绍

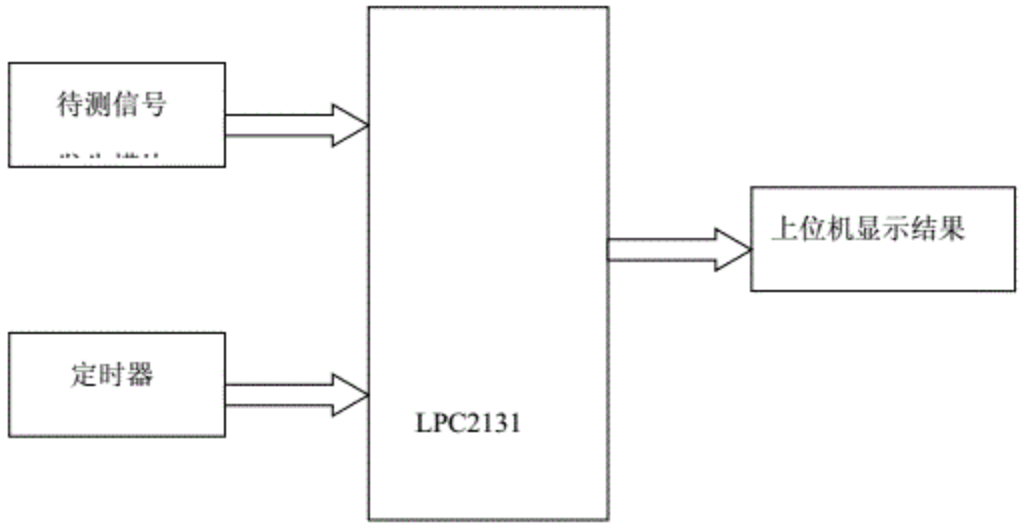


图 3.1 原理框图

由上图知，一个被测信号进入 ARM 开发板，然后经过 1S 的定时捕获得出频率值，再经由上位机显示出数值。

3.2 电源

电源模块——参考电压源为系统芯片如 A/D、D/A 转换 IC 或外设提供参考电压，电路如图 1。电压源以 TL431 为核心，RW1 调节参考电压，“电压输出 1、2”调节范围分别为 0~2.5V、5~9V。

参考电压源：“电压输出 1”0~2.5V 可调；“电压输出 2”5~9V 可调；恒流源：0~200mA 可调。

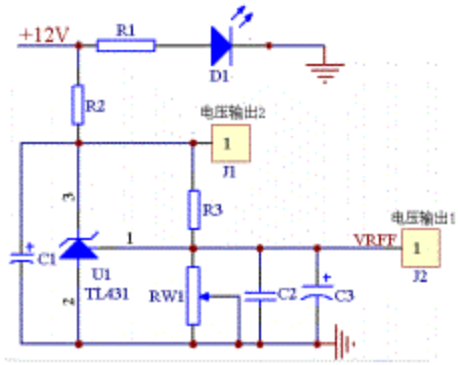


图 2 参考电压源

恒流源电路如图 3 所示，调节电位器 RW2 可以改变恒流输出，调节范围在 0~200mA 之间。

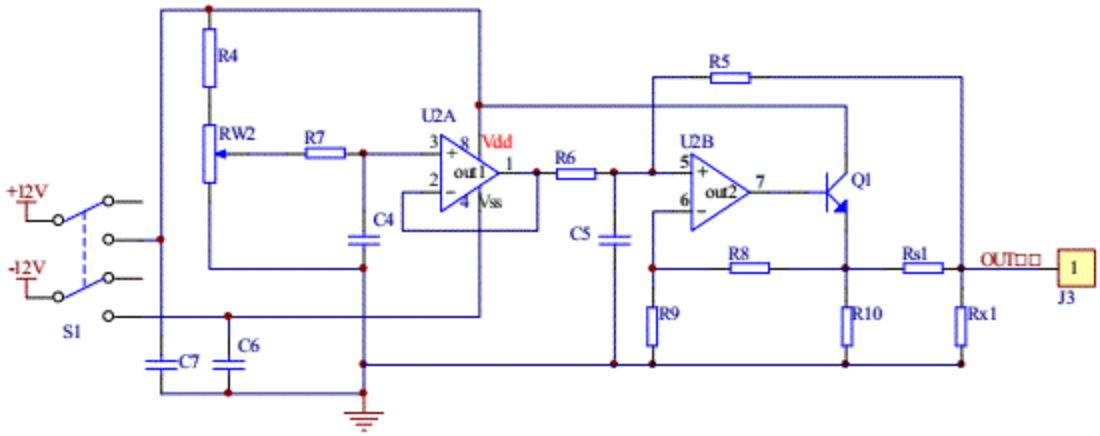
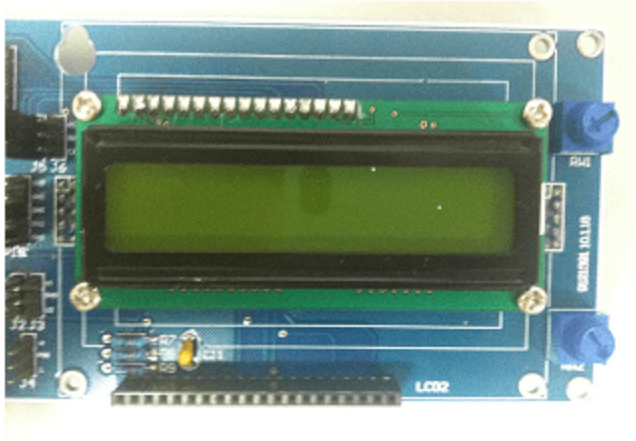


图 3 恒流源

3.3 1602 字符型 LCD

原创力文档
max.book118.com
预览与源文档一致,下载高清无水印

1602 字符型 LCD 通常有 14 条引脚线或 16 条引脚线的 LCD，多出来的 2 条线是背光电源线 VCC(15 脚)和地线 GND(16 脚)，其控制原理与 14 脚的 LCD 完全一样。



原创力文档
max.book118.com
预览与源文档一致,下载高清无水印

图3.4 LCD 实图

3.3.1 字符集

在 1602 LCD 的控制 IC 当中，字符库中 0x00~0x0F 未定义,留给使用者的自定义字符使用。但只能使用 0x00~0x07 或者 0x08~0x0F 之一。

3.3.2 显示地址

表 1 显示屏 程序显示地址

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
00H	01H	02H	03H	04H	05H	06H	07H	08H	09H	0AH	0BH	0CH	0DH	0EH	0FH
40H	41H	42H	43H	44H	45H	46H	47H	48H	49H	4AH	4BH	4CH	4DH	4EH	4FH

3.4 软件设计

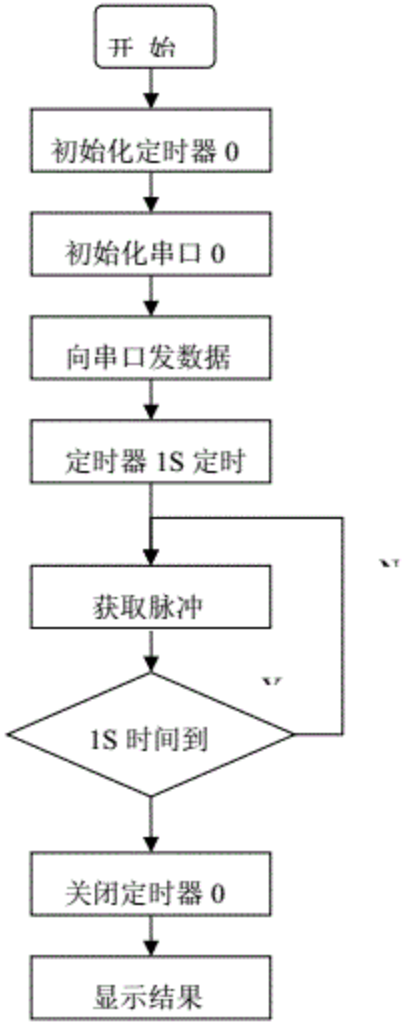


图 3.2 程序流程图

通过波形发生器产生的方波输入，定时器 T0 进行计数，通过上位机显示 T0 计数值。

四、参考文献

【1】黄智伟，税梦玲，张强，全国大学生电子设计大赛 ARM 嵌入式系统应用设计与实践【M】北京航空航天大学出版社，2011

【2】崔更申，孙安青，ARM 嵌入式系统开发与设计【M】中国电力出版社，2002

【3】周立功、陈明计、陈渝，ARM 嵌入式 Linux 系统构建与驱动开发范例【M】北京航空航天大学出版社，2006

【4】 范圣一 ， ARM原理与嵌入式系统实战【M】.机械工业出版社,2007年

【5】 陈连坤， 嵌入式系统的设计与开发【M】清华大学出版社、北京交通大学出版社,2005年

原创力文档
max.book118.com
预览与源文档一致,下载高清无水印

五、心得体会

心得体会一（雷军）

这次的课程设计是基于LPC2131的频率计，实现的主要功能用ARM的定时器/计数器的定时和计数功能，外部扩展6位LED数码管，求累计每秒进入ARM的外部脉冲个数，用LED数码管显示出来。或用上位机显示。

在做数字频率计的设计时，开始是遇到不少的问题，比如我们想如此微弱的信号是怎样被数字频率计检测的呢，频率计到底是什么设计原理呢，毕竟还没有接触过实际设计和开发，所以在考虑问题的时候往往是不全面的，也就是说这次设计还有不少的方面没有考虑周全，也一定存在着这样那样的问题。

经过这一个星期的实习，过程曲折真是一语难尽。从开始时充满激情，到最后差点有点想放弃的冲动，这之间的复杂心情，点点滴滴无不令我回味无长。特别是在设计程序的过程中，我明白到做一个好程序的不易，做一个好的编程者就更加艰难，突然就很佩服那些编程者。 这我们组一共二个人，整体上是我们二个人都在做，但个人的侧重点不同，我主要负责查询资料和设计论文及硬件方面，魏璐主要负责软件调试。通过这次课程设计，加强了我们动手、思考和解决问题的能力。在整个设计过程中主要是软件调试，这个我们花了好长时间，几乎是二个人换着调的，这个真是太麻烦了，稍微有点错误，就出现问题，这个真是对我们耐心的大挑战，不过最后还是没达到预期的效果，感觉有点遗憾。

我觉得做课程设计同时也是对课本知识的巩固和加强，平时看课本时，有时问题老是弄不懂，做完课程设计，那些问题就迎刃而解了。

认识来源于实践，实践是认识的动力和最终目的，实践是检验真理的唯一标准。所以这个期末测试之后的课程设计对我们的作用是非常大的。在课程设计的过程中，真正体会到了理论运用到实际中是具有很大的差别的。往往会出

现意想不到的问题，我们必须得做到冷静耐心的去分析问题，而不是焦躁不安，轻言放弃。当然了，这次的课程更重要的是团队的合作，我们必须得各尽所长才能解决各种困难。

心得体会二（魏璐）

我们这次的课程设计是基于LPC2131的频率计，实现的主要功能用ARM的定时器/计数器的定时和计数功能，外部扩展6位LED数码管，求累计每秒进入ARM的外部脉冲个数，用LED数码管显示出来。或用上位机显示。

当老师在之前让我们选题目时，我就开始在网上、在图书馆找资料，当接触到LPC2131频率计这个题目时，我很迷茫，不知从何下手，虽说这学期已经开ARM课了，但是自认为学的一点都不好。随后我们去上网查找资料，去图书馆查文献，但是都没有找到类似的课题，最后经过我与组员的努力，勉强有些许思绪，虽说最后我们没能很完善的做出课题，但这个过程是值得高兴地。在模拟硬件电路部分，我们查了相关的资料，其中遇到了很多的繁琐问题，但经过同学帮助都得以解决；在软件方面，我们按照书上的资料，逐步学习，逐步推敲，最终写出了部分程序，虽然功能没有完全表现出来，但是我们都很认真的去动手做了。

事实上，我们遇到的问题远不止这些，但是，无论怎样的挫折，无论怎样的想要放弃，最后都坚持了下来。有困难就查资料，有困难就请教同学，有困难就解决困难!本着这样的信念和心态，我们解决了一个个的困难，虽说结果不能达到预期结果，但从中我们也学到了很多知识，从原来不太熟悉的LPC2131、1602显示屏到最后的的每一部分都有所了解，我觉得这就是我们坚持到最后的最大成果，其实在很多事情来临时，我们不仅仅关心的是最后的结果，更重要的是拥有其中的过程。

在整个动手过程，既加深了我们对ARM的理论认识，又通过LPC2131这个很有意思的载体，实现了对ARM的应用。同时，对我而言，这次课程设计还有更重要的意义，那就是我开启了对ARM制作的兴趣，个人希望在以后的工作学习中，加强这方面的训练，多制作出自己感兴趣的ARM作品。

六、附件

程序清单：

```
/*****************************************Copyright
(c)*****************************************************
**                                    Guangzou ZLG-MCU Development Co.,LTD.
**                                           graduate school
**                                      http://www.zlgmcu.com
**
**--------------File Info-----------------------------------------------------------------------------
** File name:               main.c
** Last modified Date:    2004-09-16
** Last Version:            1.0
** Descriptions:            The main() function example template
**
**-------------------------------------------------------------------------------------------------------
** Created by:              Chenmingji
** Created date:            2004-09-16
** Version:                       1.0
** Descriptions:            The original version
**
**-------------------------------------------------------------------------------------------------------
** Modified by:
** Modified date:
** Version:
** Descriptions:
**
********************************************************************************************
*************************/
#include "config.h"
```

```
            int main (void)
{// add user source code
      return 0;
}
/*********************************************************************************
*****************************
**                                    End Of File
*********************************************************************************
*************************/
/*********************************************************************************
* 文 件 名：lihui.c
* 功      能：累计每秒进入 ARM 的外部脉冲个数，用用上位机显示。
*********************************************************************************/
#include   "config.h"
#define        UART_BPS     115200              // 定义通讯波特率
       uint8     Cout;
       uint32    PinStat;

/*********************************************************************************
* 名      称：Time0Init()
* 功      能：初始化定时器 0，定时时间为 1S。
*********************************************************************************/
void    Time0Init(void)
{     /* Fcclk = Fosc*4 = 11.0592MHz*4 = 44.2368MHz
          Fpclk = Fcclk/4 = 44.2368MHz/4 = 11.0592MHz
      */
      T0PR = 99;                                  //          设置定时器 0,分频为 100 分频，得
110592Hz
      T0MCR = 0x03;                               // 匹配通道 0 匹配中断并复位 T0TC
```

原创力文档
max.book118.com
预览与源文档一致,下载高清无水印

```
    T0MR0 = 110592;                          // 比较值(1S 定时值)
    T0TCR = 0x03;                            // 启动并复位 T0TC
    T0TCR = 0x01;                           //启动定时器 0
}
/*****************************************************************************
* 名    称：DelayNS()
* 功    能：长软件延时
* 入口参数：dly      延时参数，值越大，延时越久
*****************************************************************************/
void  DelayNS(uint32  dly)
{   uint32  i;
    for(; dly>0; dly--)
    {
        for(i=0; i<5000; i++);
    }

}
/*****************************************************************************
* 名    称：UART0_Ini()
*  功      能：初始化串口 0。设置为 8 位数据位，1 位停止位，无奇偶校验，波特率为
115200
*****************************************************************************/
void  UART0_Init(void)
{   uint16 Fdiv;

    U0LCR = 0x83;                            // DLAB = 1，可设置波特率
    Fdiv = (Fpclk / 16) / UART_BPS;      // 设置波特率
    U0DLM = Fdiv / 256;
    U0DLL = Fdiv % 256;
    U0LCR = 0x03;
```

```
}
/*****************************************************************************
* 名      称：UART0_SendByte()
* 功      能：向串口发送字节数据，并等待发送完毕。
* 入口参数：data       要发送的数据
*****************************************************************************/
void UART0_SendByte(uint8 data)
{
    U0THR = data;                                  // 发送数据
    while( (U0LSR&0x40)==0 );              // 等待数据发送完毕
}
/*****************************************************************************
* 名      称：main()
* 功      能：初始化 I/O 及定时器，然后不断计算脉冲个数。当定时时间到达时，取最终的脉冲个数并在向串口 UART0 发送最终频率。

*****************************************************************************/
int main(void)
{
     Cout=0;
    PINSEL0&=0xFFFFFFFC;        // 设置引脚连接模块，P0.0 为 GPIO
    IO0DIR &=0xFFFFFFFE;    //设置 P0.0 口方向，设置为输入
    PinStat= IO0PIN;              //从 IO0PIN 读取引脚状态
        Time0Init();                                        // 初始化定时器 0
         while(1)
    {
        while( (T0IR&0x01) == 0 );                  // 等待定时时间到
        T0IR = 0x01;                                        // 清除中断标志
       {
      if(IO0PIN&0x01==0)         //等待引脚电平变高
```

```
            {
                Cout++;
             }
             else
             {
                Cout=Cout;
             }
            }
            }

    PINSEL0 = 0x00000005;                  // 设置 I/O 连接到 UART0
    UART0_Init();
    while(1)
    {
      UART0_SendByte(Cout);
      DelayNS(10);
    }

   return(0);
}
```

原创力文档
max.book118.com
预览与源文档一致,下载高清无水印