# Z32R 开发板 智林测控技术研究所 www.the0.net

版本 1.0

深入浅出的讲解,带您轻松进入嵌入式开发的大门

Simbba 2008-5-5

前言	5
什么是嵌入式系统?	5
ARM	5
第一章 STM32 ARM 处理器	7
STM32 的优势	7
第二章 智林 Z32R 开发板	8
一、Z32R 开发板原理图	8
二、 <b>Z32R</b> 开发板板图布局	10
第三章 好用的 Keil for ARM 开发环境	11
第四章 开始行动第一个实验	13
一、   启动 Keil	13
二、 建立新项目	13
设置编译器	13
建立新项目	15
编写第一个程序	17
建立空白文档	17
第五章 GPIO 试验一:输出	21
硬件	21
软件	21
配置管脚	23
输出路径	24
连接仿真器	25
调试准备	26
开始调试	28
第六章 点亮液晶背光: PWM 试验	29
硬件	29
软件	29
第七章 点亮液晶屏	
硬件	

Z32R 开发板

软件 1:显示英文字符串31
软件 2:显示汉字
软件 3:显示位图 BMP 35
第八章 GPIO 试验二:输入
硬件
软件 1: 输入实验
软件 1: 游戏
第九章 ADC 试验 40
硬件40
软件40
第十章 PWM 输出试验:电子琴42
硬件42
软件42
第十一章 I2C 试验:读写 EEPROM 44
硬件44
软件
第十二章 USB 接口
硬件
软件
第十三章 SD 卡试验
硬件
软件
第十四章 串口通信试验
硬件
软件一: UARTO 查询方式50
软件二: UARTO 中断方式52
第十五章 CAN 总线通信试验55
硬件55
软件55
第十六章 定时器试验:使用中断方式62

05.5	
	K

第十七章 RTC 试验6
硬件6
软件6
第十八章 外部中断试验6
硬件6
软件6
第十九章 入侵检测试验7
硬件7
软件7
第二十章 看门狗试验7
软件7
第二十一章 软件中断试验7
软件7
第二十二章 实时操作系统试验: uC/OS-II7
uC/OS-II 作者致中国用户的公开信7
我们对操作系统使用的看法7
软件7

# 前言

智林测控的 **Z32R** 开发板是为了初学者设计的一款入门级开发板,考虑到趣味和实用,板 上带有一块彩色液晶屏,可以更方便地观察试验数据.

在介绍片上接口资源时,不是单纯的理论讲解,而是边做边学,把理论融入实践,使读者了 解 NXP 的 ARM 芯片的核心以及各种片上资源.

嵌入式系统是一个软硬件紧密结合的系统,研发嵌入式系统需要广泛的软硬件知识,嵌入 式开发是实践性很强的领域,除了理论,必须不断实践.我们愿意和大家分享我们的甘苦.

# 什么是嵌入式系统?

一般来说,嵌入式系统是"执行专用功能并被内部计算机控制的设备或者系统.嵌入式系统不能使用通用型计算机,而且运行的是固化的软件,用术语表示就是固件(firmware),终端用户很难或者不可能改变固件."

有技术概念的人,理解上面的定义应该没有问题,但是对非技术领域的人们来说可能存在 一些障碍.一个更简单的定义是:"嵌入式系统就是包含了用户所不知晓的计算机的设备." 但是,甚至这个定义也需要某些解释.首先,大多数非 IT 人士对计算机的概念就是一个米色的 贴着"Intel-Inside"标签的箱子.嵌入式计算机的发行量远远高于基于 Intel-x86/Pentium 的台 式 PC 的发行量.Intel 或许"inside",但是嵌入式系统却是无处不在.

人们很少会意识到他们往往随身携带了好几个嵌入式系统---手机,手表或者智能卡都嵌 有它们,而且他们在与汽车,电梯,厨房设备,电视,录像机以及娱乐系统的嵌入式系统交互时也 往往对此毫无觉察.嵌入式系统在工业机器人,医药设备,电话系统,卫星,飞行系统等领域扮演 了一个更为重要的角色.正是"看不见"这一个特性嵌入式计算机与通用 PC 计算机相区分.

### ARM

究竟什么是 ARM 呢?他是英国一家电子公司的名字,全名的意思是 Advanced RISC Machine。我来多说几句,ARM 敢为天下先,12 年前首创了 chipless 的生产模式,即该公司 既不生产芯片,也不设计芯片,而是设计出高效的 IP 内核,授权给半导体公司使用,半导体公司在 ARM 技术的基础上添加自己的设计并推出芯片产品,最后由 OEM 客户采用这些芯 片来构建基于 ARM 技术的系 统产品。ARM 的产品是 IP Core,没有任何物理意义上的硬件 或者软件实体。目前全球有 103 家巨型 IT 公司在采用 ARM 技术,20 家最大的半导体厂商中 有 19 家是 ARM 的用户,包括德州仪器,Philips,Intel 等。20 大巨头中唯一没有购买 ARM 授权的是 Intel 的老对头 AMD,而 AMD 则收购了 Alchemy 公司与之抗衡,采用的是 MIPS 结构 (有少数宽带路由器使用 MIPS 处理器,思科路由器也有使用这个牌子的)。目前面向市场的有 ARM7, ARM9, ARM9E-S, StrongARM 和 ARM10 系列。所以说 ARM 处理器有许多 OEM 厂家在生产.

ARM(Advanced RISC Machines),既可以认为是一个公司的名字,也可以认为是对一类微处理器的通称,还可以认为是一种技术的名字。

#### 智林测控技术研究所 www.the0.net

Z32R 开发板

1991 年 ARM 公司成立于英国剑桥,主要出售芯片设计技术的授权。目前,采用 ARM 技术知识产权(IP)核的微处理器,即我们通常所说的 ARM 微处理器,已遍及工业控制、消费类电子产品、通信系统、网络系统、无线系统等各类产品市场,基于 ARM 技术的微处 理器应用约占据了 32 位 RISC 微处理器 75%以上的市场份额,ARM 技术正在逐步渗入到我 们生活的各个方面。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 公司是知识产权出售公司,ARM 公司只设计内核的不生产具体的芯片.

90年代初,ARM率先推出 32 位 RISC 微处理器芯片系统 SoC 知识产权公开授权概念,从此 改变了半导体行业. ARM 通过出售芯片技术授权,而非生产或销售芯片,建立起新型的微处理 器设计,生产,和销售商业模式.更重要的是,ARM 开创了电子新纪元:采用 ARM 技术的微处理 器遍及各类电子产品,在汽车,消费娱乐,成像,工业控制,网络,存储,安保和无线等市场,ARM 技 术无处不在.

ARM 公司自 1991 年正式成立以来,在 32 位 RISC (Reduced Instruction Set Computer CPU 开发领域不断取得突破,其结构已经从 V3 发展到 V6.由于 ARM 公司自成立以来,一直以 IP(Intelligence Property)提供者的身份向各大半导体制造商出售知识产权,而自己从不介入芯片的生产销售,加上其设计的芯核具有功耗低,成本低等显著优点,因此获得众多的半导体 厂家和整机厂商的大力支持,在 32 位嵌入式应用领域获得了巨大的成功,目前已经占有 75\% 以上的 32 位 RISC 嵌入式产品市场.在低功耗,低成本的嵌入式应用领域确立了市场领导地位.现在设计,生产 ARM 芯片的国际大公司已经超过 50 多家,国内中兴通讯和华为通讯等公司 也已经购买 ARM 公司的芯核用于通讯专用芯片的设计.

目前非常流行的 ARM 芯核有 ARM7TDMI, StrongARM, ARM720T, ARM9TDMI, ARM922T, ARM940T, ARM946T, ARM966T, ARM10TDM1等.自 V5 以后,ARM 公司提供 PiccoloDSP 的芯核给 芯片设计者,用于设计 ARM+DSP 的 SOC(System On Chip)结构的芯片.此外,ARM 芯片还获得了 许多实时操作系统(Real Time Operating System)供应商的支持,比较知名的有:Windows CE, Linux, pSOS, VxWorks, Nucleus, EPOC, uCOS, BeOS 等.

随着国内嵌入式应用领域的发展,ARM 芯片必然会获得广泛的重视和应用.ARM 芯片有 多达十几种的芯核结构,70 多家芯片生产厂家,以及千变万化的内部功能配置组合.

# 第一章 STM32 ARM 处理器

STM32 系列 32 位闪存微控制器基于突破性的 ARM <u>Cortex<sup>™</sup>-M3</u> 内核,这是一款专为嵌入式应用而开发的内核。STM32 系列产品得益于 Cortex-M3 在架构上进行的多项改进,包括提升性能的同时又提高了代码密度的 Thumb-2 指令集,大幅度提高的中断响应,而且所有新功能都同时具有业界最优的功耗水平。目前 ST 是第一个推出基于这个内核的主要微控制器厂商。 STM32 系列产品的目的是为 MCU 用户提供新的自由度。它提供了一个完整的 32 位产品系列,在结合了高性能、低功耗和低电压特性的同时保持了高度的集成 性能和简易的开发特性。

## STM32 的优势

- 搭载 ARM 公司最新的、具有先进架构的 Cortex-M3 内核
- ▶ 出色的实时性能
- 优越的功效
- 高级的、创新型外设
- 最大的集成性
- •易于开发,加速了面市时间

整个产品系列具有脚到脚、外设和软件的高度兼容性,为您提供最大的灵活性。

STM32 ,最优的平台 STM32 是您最优的平台,在这一平台上可以支持多项应用。

- •从较少的存储及引脚要求到更大的需求
- •从要求高速运算到使用电池供电
  - 从单一的成本敏感型到复杂的高价值产品 整个产品系列脚到脚、外设和软件的高度兼容性为您提供完全的灵活性。不必改变您的初始设计和软件,您就可以升级到一个较高的存储容量或降级到一个较低的存储容量,也可换用不同的封装形式。

Z32R开发板

# 第二章 智林 Z32R 开发板

# 一、Z32R 开发板原理图





# 、Z32R 开发板板图布局

Z32R开发板



# 第三章 好用的 Keil for ARM 开发环境

很多电子工程师都是从 51 单片机转过来的,很熟悉 Keil 的 C51 开发环境,当看到 Keil for ARM 的开发环境时,你会发现一切是那么熟悉,我们几乎不用关心汇编语言,可以直接进入 C 语言编程。我们所要学习和了解的主要内容是熟悉片上设备的用法。



Keil uVision 调试器可以帮助用户准确地调试 ARM 器件的片内外围功能(I2C、CAN、UART、 SPI、中断、I/O 口、A/D 转换器、D/A 转换器 和 PWM 模块等功能)。ULINK USB-JTAG 转换器 将 PC 机的 USB 端口与用户的目标硬件相连(通过 JTAG 或 OCD),使用户可在目标硬件上调试 代码。通过使用 Keil uVision IDE/调试器和 ULINK USB-JTAG 转换器,用户可以很方便地编辑、 下载和在实际的目标硬件上测试嵌入的程序。

支持 Philips、Samsung、 Atmel、 Analog Devices、 Sharp、 ST 等众多厂商 ARM7 内核的 ARM 微控制器。

高效工程管理的 uVision3 集成开发环境

- Project/Target/Group/File 的重叠管理模式,并可逐级设置;
- 高度智能彩色语法显示;
- 支持编辑状态的断点设置,并在仿真状态下有效。

#### 高速 ARM 指令/外设仿真器

- 高效模拟算法缩短大型软件的模拟时间;
- 软件模拟进程中允许建立外部输入信号;
- 独特的工具窗口,可快速查看寄存器和方便配置外设;
- 支持 C 调试描述语言,可建立与实际硬件高度吻合的仿真平台;

2008

•

支持简单/条件/逻辑表达式/存储区读写/地址范围等断点。

### 多种流行编译工具选择

- Keil 高效率 C 编译器;
- ARM 公司的 RealView 编译器;
- GNU GCC 编译器;
- 后续厂商的编译器。

# 第四章 开始行动---第一个实验

在本章中我们会通过实际操作,熟悉 Keil 开发环境。所以这章会很详细的解说具体操作。



出现启动画面后



进入 Keil 开发环境, Keil 会自动加载上次开发的项目。

🕎 Vibrating 🛛 🖁	tision3		
File Edit View Pr	oject Debug Fl <u>a</u> sh Pe <u>r</u> ipherals <u>T</u> ools <u>S</u> VCS <u>W</u> indow <u>H</u> elp		
	New Project		1 m 15 m
A (11) (11) 📈 🔡	Import µVision1 Project		
Project Workspace	Open Project		
🖃 🔁 Vibrating	<u>C</u> lose Project		onfig. c
🖻 🚖 Vibratin			h is assigned by the system.
⊕- 🟦 RTX_C∢ 🦺	Components, Environment, Books		
🕑 🚠 Vibrat	Select Device for Target 'Vibrating'		
E- Driver	Remove Item		
🗈 🔛 DS2431	Options for Target 'Wibrating'		
in 📩 LCD. c			a stack overflow.
- 🖬 1pc	Build target	F7	he Kernel performance.
🛄 f6x 🛗	Rebuild all target files		
⊟ ⊞ PWM.c	Tr <u>a</u> nslate G:\XrEmbedded\WANG\Vibrating\Software\RTX_RV\RTX_Config.c		
— 🛄 lpc 🛎	1 Stop build		
• ∰ Tiner(			that will run at the same time.
🗄 🔝 WORK. 🧹	] G:\XrEmbedded\WANG\Vibrating\Software\KTX_KV\Vibrating.Uv2		,
lpc	2 G:\XrEmbedded\WANG\Vibrating\Software\RIX_ex1\KIX_ex1.Uv2		
	3 C:\Keil\ARM\RV30\RTL\Boards\Keil\MCBSTR9\Http_demo\Http_demo.Uv2		
	4 G: XIrEmbedded WANG Vibrating Soltware (AKIX_V/AKIX_V.UV2		
	5 G: \ArEmbedded \2912 \Software \Doc\stra_nttp_demo\Sika_http_demo\Http_dem G. G. \WeBele dddd \2912 \Software \LED \LED \LED \LED \LED \LED \LED \LED	0.072	
	b G: \ArEmbedded\2912\Software\Ebb.0v2		A 14 5 m <sup>2</sup> 4
	<u>1</u> G: \ArEmbedded\WaVC\Wikesting\Coftware\FITS(.0v2		
	<u>o</u> G: \ArEmbedded\WaNG\Vibrating\Software\Vibrating\vibrating\vibrating Uv2		
	<ol> <li>C. (MrEmbedded/WANG(Fibrating)Software(Fibrating(Fibrating, 002</li> <li>C. (YrEmbedded/WANG(Fibrating)Software)2132tect303/2132t303 Hy2</li> </ol>		
🖹 Files 🗮 Regs 👢			
×			
wopi			
ut Wit			•
8 H I F F Build	∫ Command 入 Find in Files /		
Create a new projec	t ULINK ARM Debugger		SCRL OVR R /W



#### 设置编译器

因为 Keil 公司已经被 ARM 公司买去了,现在我们可以在 Keil 环境中使用 ARM 的编译器了,

### Z32R开发板

R

所以在建来新项目前,我们要注意一下当前编译器是什么,点击 Project 项的

🔽 First — 礦i:	sion3	
<u>F</u> ile <u>E</u> dit <u>V</u> iew	<u>Project</u> <u>D</u> ebug Fl <u>a</u> sh Pe <u>r</u> ipherals <u>T</u> ools <u>S</u> VCS <u>W</u> indow <u>H</u> elp	
1 🖆 🚅 🖬 🎒 🐰	New Project	
S 🕐 🎢 🔏 🙀	Import µVisionl Project	
Project Workspace	Open Project	
🖃 🔁 Target 1	<u>C</u> lose Project	
⊡ — 🔄 Source G		
	Components, Environment, Books	
	Select Device for Target Target I	
	Remove Item	
	options for larget larget 1	
	(*) Build target	<b>F</b> 7
	Rebuild all target files	
	Translate G:\XrEmbedded\WANG\Vibrating\Software\RTX_RV\RTX_Config.c	
	🔏 Stop build	
掸出窗口		
Components. Environ		
Project Components F	alders/Extensions Rooks	
1.		
Project Targets:	$\square \times \not \bullet \not \bullet $ Groups: $\square \times \not \bullet \not \bullet $ <u>F</u> iles: $ \times \not \bullet \not \bullet $	
Target 1	Source Group 1 Startup.s	
let as Curren	Add Files	
	确定  取消  Defaults  帮助	
点击"Folders/E	xtensions"	
Components, Environm	ment and Books	
Project Components F	olders/Extensions Eooks	
Development Tool Fo	Default File Extensions:	
Tool Base	C:\Keil\ARK\	
BIN: C:\Keil	ARM\BIN\ Asm Source: *.s*; *.src; *.a	
INC:	Object: *. obj	
LIB:	Library: *.lib	
<u>R</u> egfile:	Document: *. txt, *. n, *. in	
-Select ARM Developm	ent Tools	
✓ Use Real⊻iew Compiler	RealView Folder: C:\Keil\ARM\BIN30\	
Use Keil CARM		
Compiler	CWI_Tecl_Profin	
Compiler	Cygnus Folder: C:\Cygnus\	

确定

取消 Defaults

帮助

勾选 Use RealView Compiler,我们使用 ARM 公司的最新发行的编译器 RealVfiew。

## 建立新项目

点击主菜单的 Projcet 项的 New Project。弹出文件对话窗:

Create New	Project		?⊠
保存在( <u>I</u> ):	🗀 1-First	* <u>1</u> € ← 💽	
文件名(N):	First		保存( <u>s</u> )
保存类型( <u>T</u> ):	Project Files (*.uv2)	·	取消

选择新项目的文件夹, 输入新项目名称, 点击保存

Select Device for Target 'Target 1'	X
CPU Vendor: Acer Labs Device: Toolset	
Description:	
确定    取消	帮助

出现设备选择窗口,找到 STMicroelectronics 分类,点击前面的加号 "+" 展开,选择我们开 发板对应的 CPU 型号,例如 STM32F103RB

/

05.5

Z32R开发板

Se	elect		ice f		Target	'Targe		×
(	CPU Vend Devi Tool Data	or: ce: set base 0 I 0 I 0 I 0 I 0 I 0 I 0 I 0 I 0 I 0 I	NXP ( LPC21 ARM PC213 PC213 PC213 PC214 PC214 PC214 PC214 PC214 PC214 PC215 PC215	(fou 148 32 34 36 38 41 42 44 46 48 34 10	nded by	Philips)	Des <u>cription:</u> ARM7TDMI-S based high-performance 32-bit RISC Microcontroll 512KB on-chip Flash ROM with In-System Programming (ISP) an Two 10bit ADCs with 14 channels, USB 2.0 Full Speed Devic Two UARTs, one with full modem interface. Two I2C serial interfaces, Two SPI serial interfaces Two 32-bit timers, Watchdog Timer, PWM unit, Real Time Clock with optional battery backup, Brown out detect circuit General purpose I/O pins. CPU clock up to 60 MHz, On-chip crystal oscillator and On-c	
		0 I 0 I 0 I	.PC221 .PC221 PC222	12 14 20		~		
-							确定 取消 帮助	

点击"确定"按钮,弹出如下对话框

μVisi	on3 🛛 🕅
?	Copy Philips LPC2100 Startup Code to Project Folder and Add File to Project ?
	是(Y) 否(N)

这是在询问是否需要把启动代码拷贝到目标文件夹并添加到项目中,我们选择"是",这时候,开发环境已经为我们建立了一个只包含启动代码的空项目

💟 First - 礦ision3		
File Edit View Project Debug Flash Periphe	erals <u>T</u> ools <u>S</u> VCS <u>W</u> indow <u>H</u> elp	
12 日日   1 日日   ユヒ   年年 4 % %		
🗇 🕮 💒 🙀 🔊 Target 1 💌	A 🚍	
Project Workspace		
E Source Group 1		
🔄 🔝 Startup. s		
Files Regs 🔛 Books 🐏 Func 🖤 Tem		
×		
Indow		
		<u> </u>
8 HILL FIRE A Command A Find in Files /		• /w
	SIRUIATION SCRL OVR R	19 //

我们看到这个项目目前只包含一个汇编文件 Startup.s 是启动代码,除非非常必要,否则我 们不必修改这个文件,我们只要写 C 语言就可以了,这是 Keil 环境做的方便之处。现在我们

开始编写第一个程序。

编写第一个程序

建立空白文档

#### 点击第一个快捷按钮"Create a new file"

Eile Edit View Project Debug Flash Peripher	als Tools SVCS Window Help	The second s	
「「「「「「「」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」			
Create a new flic Target 1	1 10		
BaTarget 1 ⇒ ∰ Source Group 1 — ∬ Startup.s			
The Rep Otors Tree.			
Meddan .			
Start a La La David Command Contin Char L			4

#### 出现一个文件编辑窗口



我们先存储这个文件,点击"存储"快捷按钮,弹出对话框

Save As		· · · · · · · · · · · · · · · · · · ·
保存在( <u>I</u> ):	1-First	▼ # 1 → ▼
<ul> <li>First.plg</li> <li>First.uv2</li> <li>Startup.s</li> </ul>		
文件名(N):	First.c	保存( <u>S</u> )
保存类型(T):	All Files (*.*)	▼ 取消

在"文件名"栏输入我们想要的名字,比如 First.c,储存。这时候这个文件并没有加入到当前项目中,我们需要手工加入。在项目窗口右击"Souce Group 1"

File Edit Vie			
	w Project Debug Flash Peripherals Tools SVCS	Mindow Help	
19660	· B B コロ ほほんろろちぬ	● ● ● ● ● ● ● ● ● ●	
	🖞 🕂 Target 1 🔍 💂 🚍		
Project Workspace	* * *		
😑 🔁 Target 1			
E 🔄 Source	Ontions for Group 'Source Group 1'		
in prai		170140\R	
	Open List File	(22148 \Example \1-First \First, c	
	Open Map File		
	Open File		
	# Rebuild target		
	🔠 Build target	F7	
	Tr <u>a</u> nslate File		
	👗 Stop build		
	New Group		
	Add Files to Group 'Source Group 1'		
	A Manage Components		
	Remove Group 'Source Group 1' and its File	IS INTERNET	<b>~</b> 1
	Include Dependencies		
Files Regs	Dooks 🔊 Func 🗮 Tem 📄 First.c		
×			
×			
nop			
at Wins			
(M1)			
<u>धू न २ भ/</u> ह	Build ( Command ), Find in Files /		• •

再点击"Add Files to Group "Source Group 1""

Add Files to Group 'Source Group 1'	?>
查找范围( <u>I</u> ): 2-First	← € 👉 💷 -
First.c	
文件名(11):	Add
文件类型(I): C Source file (*.c)	Close

在弹出的对话框中选择我们刚才新建的文件。点击"Add",再点击"Close",这时,我们可以看到项目窗口里已经多了一个我们刚才建立的文件。

智林测控技术研究所 www.the0.net

2008

Project Workspace	
🖃 🔁 Target 1	
⊡ <del>[]</del> Source Group : □ ] Startup.s □ ] First.c	1

现在我们在编辑窗口里输入一个最简单的 C 程序

🕎 First – Wision3				
Eile Edit Hiew Broject Debug Flash Peripher-	als Iools SVCS Window Help			
100000 1 100 DC FF A 33 3 8	A - + +	1 B A & B B		
이 10 11 1 1 1 K Target 1	m			
Droject Workspace - x				
Es Traget I G Source Group 1 D Startup. 6 Pirst. c	<pre>6:\Xrfabridded\Z214B\Krample\l-Fir 1</pre>	nt\Firm.c		
Tites Trep: Coloris 10 fuer_ 1 Tert_ 1	first.c			
a subject of				
8 ≤ [ ≤ [ ≥ ] ≥ ] Build & Command & Find in Files /			1	•
	Simulation	L:3 C:1	S	CRL R/W

按 Build target 快捷按钮,编译



我们看到输出窗口显示如下信息 Build target 'Target 1' assembling Startup.s... compiling First.c... linking... "First.axf" - 0 Error(s), 0 Warning(s).

这说明我们已经成功编译了这个最简单的 c 程序,虽然它什么也不做。接下来我们会做一个很简单但是有点作用的程序了,并且介绍如何调试。

# 第五章 GPIO 试验一:输出

硬件

在智林 Z32R 开发板 CPU 的 57 脚,即 PB5 上接着一个发光管 D2

			PB5 F	R34 ~~~ 470 D2 FEST LED G色0805
	U1A PA0-WKUP/USART2_CTS/ADC_IN0/TIM2_CH1_ETR PA1/USART2_RTS/ADC_IN1/TIM2_CH2 PA2/USART2_TX/ADC_IN2/TIM2_CH3 PA3/USART2_RX/ADC_IN2/TIM2_CH4 PA4/SPI1_NSS/USART2_CK/ADC_IN4 PA5/SPI1_SCK/ADC_IN5 PA6/SPI1_MISO/ADC_IN6/TIM3_CH1 PA7/SPI1_MOS/ADC_IN6/TIM3_CH2 PA8/USART1_CK/TIM1_CH2 PA8/USART1_RXTIM1_CH2 PA10/USART1_RXTIM1_CH3 PA12/USART1_RTS/CANTX / USBDM (2)/TIM1_ETR PA13/JTMS-SWDAT PA14/JTCK-SWCLK	PC15-OSC32_OUT PC14-OSC32_IN PC13-ANTI_TAMP PC12 PC11 PC10 PC5 PC5/ADC_IN15 PC4/ADC_IN14 PC3/ADC_IN13 PC2/ADC_IN13 PC2/ADC_IN13	4 3 2 53 52 51 40 39 38 37 25 24 11 10 9 9	
PB5	PA13/31D1           PB/ADC_IN9/TIM3_CH3           PB1/ADC_IN9/TIM3_CH4           PB2/B0/DC_IN9/TIM3_CH4           PB3/JTD0           PB4/JTR3T           PB5/I2C1_SCU/TIM4_CH1           PB6/I2C1_SCU/TIM4_CH1           PB6/I2C1_SCU/TIM4_CH1           PB9/TIM4_CH3           PB9/TIM4_CH4           PB11/I2C2_SCU/USART3_RX           PB12/SPI2_NSVI2C2_SMBA/USART3_CK/TIM1_BKIN           PB9/TIM4_CH4           PB13/SPI2_SCK/USART3_CTS/TIM1_CH1N           PB13/SPI2_SCK/USART3_RTS/TIM1_CH2N           PB14/SPI2_MSO/USART3_RTS/TIM1_CH2N           PB14/SPI2_NSVI2SC2_SMBA/USART3_CK/TIM1_BKIN           STM32F103R	OSC_IN OSC_OUT PD2 NRST BOOT0	5 6 54 7 60	

在这个试验中,我们会写一个程序点亮它,目的是向大家介绍 GPIO(通用输入输出)的操 作方法。

### 软件

#### Z32R 开发板

```
#define LED (1 << 5) // PB5: LED D2
/*-----*\
| Delay
                              | 延时 Inserts a delay time.
                               | nCount: 延时时间
nCount: specifies the delay time length.
\*_____*/
void Delay(vu32 nCount) {
for(; nCount != 0; nCount--);
 }
/*-----*\
MIAN ENTRY
                              L
\*-----*/
int main (void) {
 stm32_Init ();
                         // STM32 setup
 for(;;) {
 GPIOB->ODR &= ~LED;
                         // switch on LED
 Delay(2000000);
 GPIOB->ODR |= LED;
                         // switch off LED
  Delay(2000000);
  }
 }
/*-----*\
END OF FILE
\*-----*/
```

#### 配置管脚

复制我们例子中的 STM32 Init.c 文件到工程目录中,并加入到工程。打开这个文件:



点击 Configuration Wizard 标签页。出现:



展开 General purpose I/O Configuration 后,继续展开 GPIOB:GPIO port B used



在 Pin 5 used as 后选择 General Purpose Output push-pull (max speed 50MHz)

#### 输出路径

05.5

因为在编译过程中,编译器会产生很多中间文件,为了使我们的程序文件夹的文件显得不凌 乱,我们可以设置输出文件的路径,点击"目标选项"快捷按钮



弹出窗口

Z32R 开发板

智林测控技术研究所 www.the0.net

Device Target Output Listing C/C++ Asm Linker Debug Utilities NXP (founded by Philips) LPC2148 Ztal (NHZ): 12.0 Code Generation ARM-Mode 
Use Cross-Module Optimization • Operating None Start Vector Г Read/Only Memory Areas Read/Write Memory Areas >ff-chip: Start: Size: off-chip: Start: Size: NoInit ROM1 C RAM1 ROM2 RAM2 Г ROMS RAM3 on-chip: IROM1 0x0 on-chip: IRAM1 0x40000000 . 0x80000 ( 0x8000 

C

确定 取消 Defaults

IRAM2

#### 选择 Output

IROM2

Device larget output Listing C/C++ Asm Linker Debug Utilities	
clect Folder for Objects	
Debug Information	Create Batch Fi
Create HEX Fil	
Big Endian	
C Create Library: .\GPIO-OUT.LIB	
Refore Wake:	
Run User Program #1	Browse
Run User Program #2	Browse
After Make: Image: Figure Beep When Complete         Start Debugging	
Run User Program #1	Browse

点击 Select Folder for Objects...,弹出对话框,可以新建一个 output 路径,然后选择它。同 样对于 Listing 也做同样设置。然后,我们可以编译程序了。

帮助

### 连接仿真器

编译通过后,就要进入调试阶段了,连接仿真器和目标板的 JTAG 接口,打开目标板电源开 关,把ULink的USB线插入到PC机。

2008

## 调试准备

Z32R开发板

为了调试程序需要做一些必要的设置,点击"目标选项"快捷按钮,点选 Debug 选项卡

Options for Target 'Target 1'	X
Device   Target   Output   Listing   C/C++   Asm   Lin C Use Simulator Settings Limit Speed to Real-Time	Debug Utilities Use: ULINK ARM Debugger
<ul> <li>✓ Load Application at Sta: ✓ Run to main() Initialization</li> <li>Restore Debug Session Settings</li> <li>✓ Breakpoints</li> <li>✓ Toolbox</li> <li>✓ Watchpoints &amp; Pi</li> </ul>	<pre>✓ Load Application at Sta:  □ Run to main() Initialization     Edit  Restore Debug Session Settings</pre>
CPU DLL: Parameter: SARM.DLL -cLPC2100	Image: Memory Display       Driver DLL:       SARM. DLL
Dialog DLL: Parameter: DARMP.DLL -pLPC2148	Dialog DLL: Parameter: TARMF.DLL -pLPC2148
确定 耶	消 Defaults 帮助 :

选择 Use:ULINK Cortex-M3 Debugger, 然后点击 Settings, 弹出窗口

Initialization       ARI Target Driver Setup         Restore Debug Sess:       ULINK USB - JTAG Adapter         V Breakpoints &       VULINK USB - JTAG Adapter         VULINK Version:       V2.10         Device Family:       Cortex-M3         Device Family:       Cortex-M3	
Restore Debug Sess:       ULINK USB - JTAG Adapter         JTAG Device Chain         Serial No:       U0589C6E         Watchpoints &       ULINK Version:         Watchpoints &       ULINK Version:         V2.10       Device Family:         Device Family:       Cortex-M3         Cortex-M3       Cortex-M3	X
SARMCM3. DLL Farame Firmware Version: V2.02 C Manual Configuration Device Name:	Move Up Down
Di al og DLL:       Parame         DARMSTM. DLL       -pSTM         Cache Options       Oownload Options         Misc Options       Verify Code Download         Verify Code Download       Verify Code Download	

🖹 Abstract...

我们可以看到,在 JATG Device Chain 窗口里已经发现了 ARM Cortex-M3 的 CPU 核心,不要 勾选 Verify Code Download 和 Download to Flash 点击 OK 退中,回到"日标选项",点选 Utilities

点击 OK 退出,回到"目标选项",点选 Utilities

Options for Target 'Target 1'	X
Device   Target   Output   Listing   C/C++   Asm   Linker   Debug   Utilities   Configure Flash Menu Command ULINK ARM Debugger Settings   Update Target before Debuggir Init File: Edit C Use External Tool for Flash Programming	
Command: LPC210x_ISP.EXE	
确定 取消 Defaults 帮助	

点选 Use Target Driver for Flash Programming,确定是 ULINK Cortex-M3 Debugger,勾选 Update Target before Debugging,点击 Settings 按钮:

Options for Target Device   Target   Output   Configure Flash Menu (  Use Target Driver ULINK Cor Init File:	'Z32R' Listing User  C/C++  Asm Command for Flash Programming tex-M3 Debugger ▼ Setti	Linker   Debug ings   ▼ Updat	g Vtilities   .e Target before   Edit	e Debugging
C Use External Tool Command: Arguments: C Run Ir	Flash Download Setup Download Function C Erase Full Chip C Erase Sectors C Do not Erase Programming Algorithm	<ul> <li>✓ Program</li> <li>✓ Verify</li> <li>✓ Reset and Run</li> </ul>	BAM for Alg	20000000 Size: 0x0800
	Description STM32F10x 128kB Flash	Device Type On-chip Flash	Device Size 128k	Address Range 08000000H - 0801FFFFH
tract	Add	Remove	Start:	Size:

勾选 Reset and Run,这个选项可以让我们下载程序后立刻运行,方便调试。



## 开始调试



我们现在可以观察一下板子上的发光管 D2 是熄灭的,点击调试快捷按钮

我们可以执行让程序单步、执行到光标位置等等调试动作,非常方便。现在我们把光标停在 主程序开始,让程序执行到光标处,然后单步执行,可以观察发光管,可以变亮,或熄灭。

# 第六章 点亮液晶背光:PWM 试验

有了液晶屏,我们就可以很方便的观察显示数据,为了进行许多后续试验提供诸多方便。点 亮液晶的第一步是产生液晶背光。这款液晶屏的背光是 LED 构成,需要 6-7V 的电压驱动, 而开发板是 5V 供电,所以我们需要一个 DC/DC 升压电路。

硬件



这是典型的 BOOST DC-DC 变换电路,由 CPU 产生的 PWM 脉冲控制三极管实现升压,会把输入的 5 伏电压提升到 6-7 伏,用来点亮液晶背光。

### 软件

在这个实验里,不需要我们写代码,只要配置一下就有了,和上个实验一样,我们配置 STM32\_Init.c 文件,使用 Configuration Wizard,我们配置 TIM2 如下图:

Z32R开发板

Expand All Collapse All Help		
Dption	Value	~
E System Timer Configuration	. <u>I.</u>	
🗄 Real Time Clock Configuration		
Timer Configuration		
😟 TIM1 : Timer 1 enabled		
🖃 TIM2 : Timer 2 enabled		
-TIM2 period [us]	100	
TIM2 detailed settings		
TIM2.PSC: Timer 2 Prescaler	0	
- TIM2. ARR: Timer 2 Auto-reload	72	
표 Timer 2 Control Register 1 Configuration (TIM2	2_CR1)	
🛨 Timer 2 Control Register 2 Configuration (TIM2	2_CR2)	
🛨 Timer 2 Slave mode control register Configurat	tion (TIM2_SMC)	
🛨 Channel 1 Configuration		
🛨 Channel 2 Configuration		
🛨 Channel 3 Configuration		
+ Channel 4 Configuration		
TIM2 interrupts	F	
. TIM3 : Timer 3 enabled		~
1		

软件运行后可以看到液晶的白色背光发光管点亮了。

# 第七章 点亮液晶屏

智林 Z32R 开发板配置了一块彩色液晶屏,驱动芯片是 SPFD54124B.

硬件



### 软件1:显示英文字符串

```
我们首先定义了底层硬件操作:
```

/*.		*\
, 	HARDWARE DEFINITIONS	Ì
\*·		*/
/*	PINS:	
	- DB0 = PC0	
	- DB1 = PC1	
	- DB2 = PC2	
	- DB3 = PC3	
	- DB4 = PC4	
	- DB5 = PC5	
	- DB6 = PC6	
	- DB7 = PC7	
	- RD = PC8	
	- RW = PC9	

	Z32R 开发板	图林测控技术研究所 www.the0.net
	- RS = PC10	
	- RST = PC11	
	- CS = PC12	*/
#de	fine PIN_CS	( 1 << 12)
#de	fine PIN_RST	( 1 << 11)
#de	fine PIN_RS	( 1 << 10)
#de	fine PIN_RW	( 1 << 9)
#de	fine PIN_RD	( 1 << 8)
#de	fine PINS_CTRL	(0x1F << 8)
#de	fine PINS_DATA	(0xFF << 0)
#de	fine PINS_ALL	(PINS_CTRL   PINS_DATA)
/*	Frahla Clask for marink	the second se
/*	Enable Clock for periphe	eral driving LCD pins */
#ae	TINE LCD_CLOCK_EN	(RCC->APBZENR  = (1 << 4)); // enable clock for GPIO
/*	nin (S setting to 0 on	1 */
/ ·	fine LCD (S(x)	$CDTOC_{ODP} = (CDTOC_{ODP} & DTN(CS)   (x ) DTN(CS)$
#ue	nin PST setting to $\theta$ on	4/10C->0DK = (GP10C->0DK & ~P1N_CS)   (X ! P1N_CS .
/*	pin KST Setting to 0 or	
#ue	rin DC cotting to 0 on 7	4
/*	pin KS setting to 0 or .	
#ae	tine LCD_RS(X)	GPIOC->ODR = (GPIOC->ODR & ~PIN_RS)   (X ? PIN_RS :
/*	pin RW setting to 0 or 1	1 */
#de	tine LCD_RW(x)	GPIOC->ODR = (GPIOC->ODR & ~PIN_RW)   (x ? PIN_RW :
/*	pin RD setting to 0 or 1	1 */
#de	fine LCD_RD(x)	GPIOC->ODR = (GPIOC->ODR & ~PIN_RD)   (x ? PIN_RD :
/*	Reading DATA pins	*/
#de	fine LCD DATA IN	(((GPIOC->IDR & PINS DATA) >> 0) & 0xFF)
/*	Writing value to DATA p	ins */
#de	fine LCD DATA OUT(x)	GPIOC->ODR = (GPIOC->ODR & ~PINS DATA)   (x << 0):
	///	
/*	Setting all pins to out	put mode */
#de	fine LCD ALL DIR OUT	GPIOC->CRL = (GPIOC->CRL & 0x00000000)   0x33333333
		GPIOC->CRH = (GPIOC->CRH & 0xFFF00000)   0x00033333;
/*	Setting DATA pins to in	put mode */
#de	fine LCD_DATA_DIR_IN	GPIOC->CRL = (GPIOC->CRL & 0x00000000)   0x44444444
/*	Setting DATA pins to out	tput mode */

我们用 GPIO 模拟驱动液晶的总线时序。

```
LCD_CS(0);
 LCD_RD(1);
 LCD_RS(1);
 LCD_Bus_Write_Byte( Data );
 LCD_CS(1);
 }
void LCD_Reg_Set(unsigned int Cmd, unsigned int Data) {
 LCD_CS(0);
 LCD_RD(1);
 LCD_RS(0);
 LCD_Bus_Write_Byte( Cmd );
 LCD_RS(1);
 LCD_Bus_Write_Byte( Data );
 LCD_CS(1);
 }
这两个函数用来写指令和数据。
   剩下的工作,就是按照 SPFD54124B 的手册进行操作,大家直接读源码就可以理解了,
这里就不罗嗦了。
   本节的例子程序的运行结果是在液晶上显示几行英文文字。
int main (void) {
 stm32_Init ();
                                          // STM32 setup
 LCD_Init();
 LCD_Clear_Screen(Blue);
 for(;;) {
                                           // switch on LED
   GPIOB->ODR &= ~LED;
   Delay(2000000);
   GPIOB->ODR |= LED;
                                           // switch off LED
   Delay(2000000);
   Font = 0;
   LCD_PutString(10,30,"STM32F EVAL BOARD",Cyan,Blue);
   Font = 1;
   LCD_PutString(38,46,"Version 1.0",Green,Blue);
   Font = 0;
   LCD_PutString(30,65,"ZERO Research",Yellow,Blue);
   LCD_PutString(36,81,"www.the0.net",Magenta,Blue);
   }
  }
```

## 软件2:显示汉字

我们可以定义一个所用汉字	勺小字库,文件为 GB1616.h,其结构如下:	
//	汉字字模的数据结构定义 //	
struct typFNT_GB16	// 汉字字模数据结构	

```
{
       unsigned char Index[3];
                                              // 汉字内码索引
                                           // 点阵码数据
       char Msk[32];
};
然后定义字模数据:
const struct typFNT_GB16 codeGB_16[] =
                                      // 数据表
{
"智", 0x00,0x00,0x18,0x00,0x11,0x22,0x1F,0xBF,0x64,0x22,0x05,0x22,0x7F,0xA2,0x0A,0x3E,
    "林", 0x00,0x00,0x0C,0x18,0x08,0x10,0x0A,0x12,0x7F,0x7F,0x08,0x10,0x18,0x30,0x18,0x58,
    0x2C,0x54,0x2A,0x54,0x4A,0x92,0x08,0x91,0x09,0x10,0x08,0x10,0x08,0x10,0x08,0x10,
"测", 0x00,0x00,0x20,0x02,0x1B,0xE2,0x0A,0x2A,0x02,0xAA,0x42,0xAA,0x2A,0xAA,0x2A,0xAA,
    0x0A,0xAA,0x12,0xAA,0x12,0xAA,0x72,0xAA,0x10,0x82,0x11,0x42,0x12,0x2A,0x14,0x24,
"控", 0x00,0x00,0x18,0x30,0x11,0x22,0x11,0xFF,0x7D,0x52,0x12,0x54,0x10,0x91,0x14,0x8F,
    0x19,0x00,0x70,0x04,0x51,0xFE,0x10,0x20,0x10,0x20,0x10,0x22,0x73,0xFF,0x20,0x00,
.. ...
};
接着我们来做一个显示汉字的函数:
#include "GB1616.h" //16*16 汉字字模
void PutGB1616(unsigned short x, unsigned short y, unsigned char c[2],
unsigned int f,unsigned int b){
    unsigned int i,j,k;
    LCD SetArea(x, y, x+16-1, y+16-1);
    LCD_Inst(0x2C);
    IO1DIR |= LCD_BUS_DAT;
    IO0CLR = LCD BUS CS;
    IO0SET = LCD_BUS_RD;
    IO1SET = LCD_BUS_RS;
    for (k=0;k<49;k++) {</pre>
      if ((codeGB_16[k].Index[0]==c[0])&&(codeGB_16[k].Index[1]==c[1])){
       for(i=0;i<32;i++) {</pre>
          unsigned short m=codeGB_16[k].Msk[i];
         for(j=0;j<8;j++) {</pre>
            if((m&0x80)==0x80) {
               LCD_Bus_Write_Byte(f>>8);
               LCD_Bus_Write_Byte(f);
               }
            else {
```

```
LCD Bus Write Byte(b>>8);
```

```
LCD_Bus_Write_Byte(b);

}

m<<=1;

}

}

IO0SET = LCD_BUS_CS;

IO1DIR &= ~LCD_BUS_DAT;

}
```

```
我们可以调用这个函数显示汉字了:
#include "TFT018.h"
```

```
int main() {
   BacklightOn();
   LCD_Init();
   LCD_Clear_Screen(Blue);
   for(;;) {
       Font = 0;
       LCD_PutString(10,0,"STM32 开发板",Cyan,Blue);
       LCD_PutString(0,16,"STM32 Dev Board",Cyan,Blue);
       Font = 1;
       LCD_PutString(38,46,"Version 1.0",Green,Blue);
       Font = 0;
       LCD_PutString(10,60,"智林测控技术研究所",Yellow,Blue);
       LCD_PutString(30,80,"ZERO Research",Yellow,Blue);
       LCD_PutString(36,100,"www.the0.net",Magenta,Blue);
       }
   }
```

### 软件 3:显示位图 BMP

这里我们使用了 ARM 汇编器的一个功能,就是可以直接引入二进制文件,我们在 BMP.s 汇 编文件中引入:

```
PRESERVE8
; ------ Load bmp picture ------
AREA |subr|,DATA,READONLY
EXPORT bmp
bmp
; incbin seaside.BMP
; incbin tree.BMP
; incbin tree.BMP
; incbin beauty.BMP
```

	Z32R <del>J</del>	F发板	智林测控技术研究所、	www.the0.net	
;	incbin girl.BMP				
;	incbin sunflower	r.BMP			
	incbin boat.BMP				
;	incbin lake.BMP				
;	incbin sweetdays	s.BMP			
	END				
イトノ			LE TTELLEU		

我们准备好了几个位图,大家可以把分号去掉,看看别的图片,但是每次只能有一个图片打 开,其他的前面都要加上分号注释掉。

05.5
# 第八章 GPIO 试验二:输入

智林 Z32R 开发板带有一个游戏杆和 2 个按钮开关作为开关量输入试验用。

硬件





智林测控技术研究所 www.the0.net

#### 05.5

## 软件1:输入实验

Z32R 开发板

```
当然,我们需要在 STM32_Init.c 的 Configuration Wizard 中配置相应的端口为输入。
int main (void) {
  stm32_Init ();
                                                    // STM32 setup
  LCD_Init();
  LCD_Clear_Screen(Blue);
  for(;;) {
    Font = 0;
    LCD_PutString(30,0,"STM32F 开发板",Cyan,Blue);
    LCD_PutString(30,16,"GPIO 输入演示",Red,Yellow);
    Font = 1;
    LCD PutString(38,32,"Version 1.0",Green,Blue);
    Font = 0;
    LCD_PutString(10,94,"智林测控技术研究所",Yellow,Blue);
    LCD_PutString(36,111,"www.the0.net",Magenta,Blue);
    if (((GPIOB->IDR & JOYSTICK) == JOYSTICK )) { // Check if JOYSTICK key is pressed
      GPIOB->ODR &= ~LED;
                                                         // switch off LED
      }
    else {
                                                        // JOYSTICK key is pressed
                                                         // switch on LED
      GPIOB->ODR |= LED;
      }
    if (((GPIOB->IDR & UP) == 0 ))
                                                // Check if UP is pressed
      LCD_PutString(48,40,"上",Yellow,Red);
    else
      LCD_PutString(48,40,"上",Red,Yellow);
    if (((GPIOB->IDR & DOWN) == 0 ))
                                                 // Check if DOWN is pressed
      LCD_PutString(48,76,"下",Yellow,Red);
    else
      LCD_PutString(48,76,"下",Red,Yellow);
    if (((GPIOB->IDR & LEFT) == 0 ))
                                               // Check if LEFT is pressed
      LCD PutString(30,58,"左",Yellow,Red);
    else
      LCD_PutString(30,58,"左",Red,Yellow);
    if (((GPIOB->IDR & RIGHT) == 0 ))
                                                // Check if RIGHT is pressed
      LCD_PutString(66,58,"右",Yellow,Red);
    else
      LCD_PutString(66,58,"右",Red,Yellow);
    if (((GPIOB->IDR & OK) == 0 ))
                                                // Check if OK is pressed
      LCD_PutString(48,58,"OK",Yellow,Red);
    else
```



这段程序运行后,按下游戏杆的任何一个按键,发光管都会发光,在液晶上有相应的位置变成红色。

软件1:游戏

有了液晶显示和输入做基础我们可以写个小游戏:贪吃蛇。大家自己看看吧。

Z32R开发板

## 第九章 ADC 试验

硬件



需要短接跳线 JP1 的 2、3 脚。我们读取 PA1、2、3 脚的电压。

软件

我们通过 DMA 读取 ADC 的结果,初始化如下 void adc\_Init (void) { // GPIOA->CRL &= ~0x0000000F; /\* set PIN1 analog input (see stm32\_Init.c) \*/ RCC->AHBENR |= (1<<0); /\* enable periperal clock for DMA \*/

```
2008
```

```
DMA_Channel1->CMAR = (unsigned long)&analog; /* set chn1 memory address
                                                                                 */
 DMA_Channel1->CPAR = (unsigned long)&(ADC1->DR);  /* set chn1 peripheral address
*/
 DMA_Channel1->CNDTR = 3;
                                             /* transmit 3 words
                                                                                     */
// DMA_Channel1->CCR = 0x00002520;
                                                          /* configure DMA channel 1
*/
 DMA_Channel1->CCR = 0x000025A0;
                                             /* configure DMA channel 1
                                                                                     */
                                            /* circular mode, memory increment mode */
                                            /* memory & peripheral size 16bit
                                                                                     */
                                            /* channel priotity high
                                                                                    */
 DMA_Channel1->CCR |= (1 << 0);</pre>
                                            /* enable DMA Channe1
                                                                                     */
 RCC->APB2ENR |= (1<<9);</pre>
                                            /* enable periperal clock for ADC1
                                                                                     */
 ADC1 - SQR1 = 0 \times 00200000;
                                                                                     */
                                            /* three conversions
 ADC1->SQR3 = (3<<10) | (2<<5) | (1<<0); /* set order to chn1 - chn2 - chn3
                                                                                     */
 ADC1->SMPR2 = (5<< 9) | (5<<6) | (5<<3);
                                            /* set sample time (55,5 cycles)
                                                                                     */
 ADC1 - >CR1 = 0 \times 00000100;
                                             /* use independant mode, SCAN mode
                                                                                     */
                                                                                     */
                                            /* data align right, cont. conversion
 ADC1 - > CR2 = 0 \times 000 E0103;
                                            /* EXTSEL = SWSTART
                                                                                    */
                                            /* enable ADC, DMA mode
                                                                                    */
 ADC1->CR2 |= 0x00500000;
                                              /* start SW conversion
                                                                                     */
 }
```

软件运行后,我们调节电位器 R7,在液晶屏上可以看到电压显示。

## 第十章 PWM 输出试验:电子琴

硬件



SOUND 接到 CPU 的 PBO 上, 即定时器 3 的第 3 通道:



#### 软件

在上一章的基础上我们通过电位器的电压控制扬声器的频率,以下是程序的主要代码:

TIM3->ARR = i;

#### TIM3->CCR3 = i/2;

这里控制了输出频率。当然,我们需要在 STM32\_Init.c 的 Configuration Wizard 中配置 PWM,

智林测控技术研究所 www.the0.net

2008

大家自己打开例子看吧。

Z32R开发板

智林测控技术研究所 www.the0.net

# 第十一章 I2C 试验:读写 EEPROM

硬件



### 软件

采用查询方式的 I2C 子程序:	
/*	*\
SOFTWARE DATA	
\*	*/
#define I2C_Speed 100000	
#define I2C_OwnAddr 0xA0	
I2C_InitTypeDef i = {	
I2C_Mode_I2C,	
I2C_DutyCycle_2,	
I2C_OwnAddr,	
I2C_Ack_Enable,	
I2C_AcknowledgedAddress_7bit,	
I2C_Speed	
};	
/*	*\
I2C Initialisation	
\*	*/

Z32R 开发板

2008

```
void I2C_Initialisation( void ) {
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_AFI0, ENABLE);
  /* Enable I2C1 clocks */
  RCC_APB1PeriphClockCmd(RCC_APB1Periph_I2C1, ENABLE);
  I2C_DeInit(I2C1);
  I2C_Init(I2C1, &i);
  I2C_Cmd (I2C1, ENABLE);
  }
u32 waitForEvent(u32 event) {
  int i;
  for (i = 2000000; i > 0; i--) {
    u32 \ s = I2C1 -> SR1;
    //if (s & (I2C FLAG TIMEOUT | I2C FLAG PECERR | I2C FLAG OVR | I2C FLAG AF |
I2C_FLAG_ARLO | I2C_FLAG_BERR | I2C_FLAG_STOPF));
    if (s & event) return s;
    }
 return 0;
  }
void sendStart(void) {
    I2C_GenerateSTART(I2C1, ENABLE);
                                                                  // send start when bus
becomes available
    waitForEvent(I2C FLAG SB);
    I2C1 \rightarrow DR = I2C OwnAddr;
                                                                       11
                                                                           send slave
address for transmission
    waitForEvent(I2C_FLAG_ADDR);
    I2C1 \rightarrow SR2;
                                                                  // clear event
 }
void sendData(u8 data) {
  //waitForEvent(I2C FLAG TXE);
  waitForEvent(I2C_FLAG_BTF);
  I2C1 \rightarrow DR = data;
                                                                  // send byte
  }
void sendStop(void) {
  waitForEvent(I2C_FLAG_TXE);
 I2C_GenerateSTOP(I2C1, ENABLE);
                                                              // send stop after current
byte
 }
u8 receiveByte(void) {
  I2C GenerateSTART(I2C1, ENABLE);
                                                                  // send start when bus
becomes available
  waitForEvent(I2C_FLAG_SB);
  I2C_AcknowledgeConfig(I2C1, DISABLE);
                                                              // only one byte will be
read
```

```
Z32R开发板
                                          智林测控技术研究所 www.the0.net
05.5
         I2C1 \rightarrow DR = I2C_0wnAddr | 0x01;
                                                                 // send slave address
       for reception
         waitForEvent(I2C_FLAG_ADDR);
         I2C1\rightarrowSR2;
                                                             // clear event
         waitForEvent(I2C_FLAG_RXNE);
         I2C GenerateSTOP(I2C1, ENABLE);
                                                             // send stop after current
       byte
         return I2C1->DR;
                                                                 // receive byte
         }
                        -----*\
       /*---
        | I2C Write Byte
       \*-----
                                                              */
       void WriteByte(u16 addr, u8 data) {
         /* Enable I2C1 acknowledgement if it is already disabled by other function */
         //I2C_AcknowledgeConfig(I2C1, ENABLE);
         //I2C_AcknowledgeConfig(I2C1, DISABLE);
                                                             // only one byte will be
       read
         sendStart();
         sendData( addr & 0xFF );
         //I2C_AcknowledgeConfig(I2C1, DISABLE);
                                                            // only one byte will be
       read
         sendData( data );
         waitForEvent(I2C FLAG BTF);
         sendStop();
         }
       /*-----
                                                              -*\
        I2C Read Byte
       \*-----
                                                              */
       u8 ReadByte(u16 addr) {
         /* Enable I2C1 acknowledgement if it is already disabled by other function */
         //I2C_AcknowledgeConfig(I2C1, ENABLE);
         sendStart();
         sendData( addr & 0xFF );
         //sendStart();
         //sendStop();
         return receiveByte();
         }
       /*-----
                                                              *\
        Delay
        延时 Inserts a delay time.
        nCount: 延时时间
```

```
nCount: specifies the delay time length.
\*-----
                                                           */
void Delay(vu32 nCount) {
 for(; nCount != 0; nCount--);
 }
/*-----
                                                           *\
 MIAN ENTRY
\*-----
                                                           -*/
int main (void) {
 char s[20];
 stm32_Init ();
                                             // STM32 setup
 LCD_Init();
 I2C Initialisation();
 LCD Clear Screen(Blue);
 Font = 0;
 LCD PutString(30,0,"STM32F 开发板",Cyan,Blue);
 LCD_PutString(10,113, "智林测控技术研究所", Yellow, Blue);
 LCD_PutString(40, 20, "I2C Test", Blue, Cyan);
 LCD_PutString(5,38,"24C02 Address 0x00",Cyan,Blue);
 LCD PutString (4, 62, "Write:0x55", Red, Yellow);
 WriteByte (0x00, 0x55);
 Delay(30000);
 sprintf(s, "Read:0x%2X", ReadByte(0x00) );
 LCD_PutString(4, 82, s, Red, Yellow);
 for(;;) {
   GPIOB->ODR &= ~LED;
                                             // switch on LED
   Delay(200000);
   GPIOB \rightarrow ODR \mid = LED;
                                             // switch off LED
   Delay(2000000);
   }
 }
/*-----
                                                           *\
END OF FILE
\*-----
                                                           -*/、
```

程序运行后,我们可以在液晶屏上看到写入 0x55,读出 0x55。

Z32R开发板

智林测控技术研究所 www.the0.net

# 第十二章 USB 接口

STM32F103 系列处理器带有片上 USB2.0 设备接口,硬件连接非常简单。

硬件



做 USB 试验时,要短接跳线 JP3 的 2、3。

### 软件

USB 软件很复杂,好在 Keil 公司提供了可以立即投入使用的协议栈,例子程序模拟了一个 U 盘,上电后,插入 USB 口,Windows 自动识别。

# 第十三章 SD 卡试验

硬件



#### 软件

这个例子里我们使用了开源文件系统 FatFS (http://elm-chan.org/fsw/ff/00index\_e.html) 来操 作 SD 卡。

```
看主要代码:
f_mount(0, &fs);
fres = f_open (&F, filename, FA_READ | FA_WRITE | FA_CREATE_ALWAYS);
//printf("status f_open: %d\n\r", fres); //status 0 = success
f_sync (&F);
fres = f_write (&F, buf1, sizeof (buf1) - 1, &bytesWritten);
f_sync (&F);
//printf("status f_write: %d\n\r", fres); //status 0 = success
f_close (&F);
f_mount(0, NULL);
我们建立了一个文件并写入一段文字, 我们插入 SD 卡, 程序运行后, 就会多了一个 test.txt
的文件, 我们可以用读卡器在 PC 机上看到。
```

# 第十四章 串口通信试验

硬件



## 软件一:UART0 查询方式

用串口线连接 J8(UARTO)到 PC 机串口,打开超级终端,设置波特率为 9600,程序运行后我 们可以键入字符,板子会反馈我们的字符。首先设置如下:

Expand All Collapse All	Help	
Dption	Value	
Clock Configuration	<u> <u> </u></u>	
-Independent Watchdog Configuration		
-System Timer Configuration		
-Real Time Clock Configuration		
Timer Configuration		
USART Configuration		
🚍 USART1 : USART #1 enable		
Baudrate	9600 Baud	
- Data Bits	8 Data Bits	
- Stop Bits	1 Stop Bit	
- Parity	No Parity	
- Flow Control	None	
- Pins used	TX = PA9, RX = PA10	
⊡-USART1 interrupts		
🕀 USART2 : USART #2 enable		
⊡–USART3 : USART #3 enable		
-External interrupt/event Configuration		
-Tamper Configuration		
General purpose I/O Configuration		

/*	*\
SendChar	1
Write character to Serial Port.	
\*	*/
<pre>int SendChar (int ch) {</pre>	

while (!(USART1->SR & USART\_FLAG\_TXE));

智林测控技术研究所 www.the0.net

```
2008
```

```
USART1->DR = (ch & 0x1FF);
 return (ch);
}
   */
| GetKey
                                       Read character to Serial Port.
\*_____*/
int GetKey (void) {
 while (!(USART1->SR & USART FLAG RXNE));
 return ((int)(USART1->DR & 0x1FF));
}
/*-----*\
MIAN ENTRY
                                       \*_____*/
int main (void) {
 stm32_Init ();
                              // STM32 setup
 LCD_Init();
 LCD_Clear_Screen(Blue);
 Font = 0;
 LCD_PutString(30,0,"STM32F 开发板",Cyan,Blue);
 LCD_PutString(15,16,"UART 查询方式通信",Red,Yellow);
 Font = 1;
 LCD_PutString(38,32,"Version 1.0",Green,Blue);
 Font = 0;
 LCD_PutString(10,94,"智林测控技术研究所",Yellow,Blue);
 LCD_PutString(36,111,"www.the0.net",Magenta,Blue);
 printf ("Polling mode Serial I/O Example\r\n\r\n");
 for(;;) {
  unsigned char c;
  printf ("Press a key. ");
  c = getchar ();
  printf ("\r\n");
  printf ("You pressed '%c'.\r\n\r\n", c);
  }
 }
/*_____*\
| END OF FILE
                                       \*-----*/
```

#### 05.5

#### 软件二:UARTO 中断方式

Z32R开发板

```
/*-----*\
USART1_IRQHandler
                                                  | Handles USART1 global interrupt request.
                                                    \*-----*/
void USART1_IRQHandler (void) {
 volatile unsigned int IIR;
 struct buf_st *p;
  IIR = USART1->SR;
  if (IIR & USART_FLAG_RXNE) { // read interrupt
  USART1->SR &= ~USART_FLAG_RXNE; // clear interrupt
    p = \&rbuf;
    if (((p->in - p->out) & ~(RBUF_SIZE-1)) == 0) {
      p->buf [p->in & (RBUF_SIZE-1)] = (USART1->DR & 0x1FF);
      p->in++;
    }
   }
   if (IIR & USART_FLAG_TXE) {
    USART1->SR &= ~USART_FLAG_TXE; // clear interrupt
    p = \&tbuf;
    if (p->in != p->out) {
      USART1->DR = (p->buf [p->out & (TBUF_SIZE-1)] & 0x1FF);
      p->out++;
      tx_restart = 0;
    }
    else {
      tx_restart = 1;
       USART1->CR1 &= ~USART_FLAG_TXE; // disable TX interrupt if nothing
to send
    }
   }
}
/*-----
                                                 ----*\
| buffer_Init
                                                  | initialize the buffers
```

```
\*-----*/
void buffer_Init (void) {
 tbuf.in = 0;
                                 // Clear com buffer indexes
 tbuf.out = 0;
 tx_restart = 1;
 rbuf.in = 0;
 rbuf.out = 0;
}
/*-----*\
| SendChar
                                        | transmit a character
                                        \*-----
                                        ----*/
int SendChar (int c) {
 struct buf_st *p = &tbuf;
                                // If the buffer is full, return an error
value
 if (SIO_TBUFLEN >= TBUF_SIZE)
 return (-1);
 p->buf [p->in & (TBUF_SIZE - 1)] = c; // Add data to the transmit buffer.
 p->in++;
 if (tx_restart) {
                                // If transmit interrupt is disabled,
enable it
 tx_restart = 0;
  USART1->CR1 |= USART_FLAG_TXE;
                            // enable TX interrupt
 }
 return (0);
}
/*-----*\
| GetKey
                                        | receive a character
                                        \*-----*/
int GetKey (void) {
 struct buf_st *p = &rbuf;
 if (SIO_RBUFLEN == 0)
  return (-1);
```

#### 05.5

#### Z32R开发板

```
return (p->buf [(p->out++) & (RBUF_SIZE - 1)]);
}
*\_____*
| Delay
                                         | 延时 Inserts a delay time.
                                          Τ
| nCount: 延时时间
                                         nCount: specifies the delay time length.
\*-----*/
void Delay(vu32 nCount) {
 for(; nCount != 0; nCount--);
 }
/*-----*\
MIAN ENTRY
                                         \*-----*/
int main (void) {
 buffer_Init();
                    // init RX / TX buffers
 stm32_Init ();
                                // STM32 setup
 LCD_Init();
 LCD_Clear_Screen(Blue);
 Font = 0;
 LCD_PutString(30,0,"STM32F 开发板",Cyan,Blue);
 LCD_PutString(15,16,"UART中断方式通信",Red,Yellow);
 Font = 1;
 LCD_PutString(38,32,"Version 1.0",Green,Blue);
 Font = 0;
 LCD_PutString(10,94,"智林测控技术研究所",Yellow,Blue);
 LCD_PutString(36,111,"www.the0.net",Magenta,Blue);
 printf ("Interrupt driven Serial I/O Example\r\n\r\n");
 for(;;) {
  unsigned char c;
  printf ("Press a key. ");
  c = getchar ();
  printf ("\r\n");
  printf ("You pressed '%c'.\r\n\r\n", c);
  }
 }
/*_____*\
END OF FILE
```

\\*-----\*/

## 第十五章 CAN 总线通信试验

硬件



#### 软件

```
CAN 的底层操作实现在 CAN.c 中:
#include <stm32f10x lib.h>
                                         // STM32F10x Library Definitions
#include "STM32_Reg.h"
                                          // STM32 register and bit Definitions
#include "STM32_Init.h"
                                          // STM32 Initialization
#include "CAN.h"
                                          // STM32 CAN adaption layer
CAN_msg
                                          // CAN messge for sending
           CAN_TxMsg;
CAN_msg
           CAN_RxMsg;
                                         // CAN message for receiving
unsigned int CAN_TxRdy = 0;
                                         // CAN HW ready to transmit a message
unsigned int CAN_RxRdy = 0;
                                         // CAN HW received a message
/*-----
                    -----
 setup CAN interface
*_____
                   ----*/
void CAN_setup (void) {
 unsigned int brp = stm32_GetPCLK1();
 RCC->APB1ENR |= RCC APB1ENR CANEN;
                                          // enable clock for CAN
                                        // Note: MCBSTM32 uses PB8 and PB9 for CAN
 RCC->APB2ENR |= RCC_APB2ENR_AFIOEN;
                                          // enable clock for Alternate Function
 AFIO->MAPR &= 0xFFFF9FFF;
                                          // reset CAN remap
 AFIO->MAPR
            = 0 \times 00004000;
                                          // set CAN remap, use PB8, PB9
 RCC->APB2ENR |= RCC_APB2ENR_IOPBEN;
                                         // enable clock for GPIO B
 GPIOB->CRH &= ~(0x0F<<0);</pre>
```

```
Z32R开发板
                                智林测控技术研究所 www.the0.net
 GPIOB->CRH |= (0x08<<0);
                                       // CAN RX pin PB.8 input push pull
 GPIOB->CRH &= \sim(0 \times 0 F <<4);
 GPIOB->CRH |= (0x0B<<4);</pre>
                                       // CAN TX pin PB.9 alternate output push
pull
 NVIC->Enable[0] |= (1 << (USB HP CAN TX IRQChannel & 0x1F));// enable interrupt
 NVIC->Enable[0] |= (1 << (USB_LP_CAN_RX0_IRQChannel & 0x1F));// enable interrupt</pre>
 CAN->MCR = (CAN_MCR_NART | CAN_MCR_INRQ); // init mode, disable auto.
retransmission
                                      // Note: only FIFO 0, transmit mailbox 0
used
 CAN->IER = (CAN_IER_FMPIE0 | CAN_IER_TMEIE); // FIFO 0 msg pending, Transmit mbx
empty
 /* Note: this calculations fit for PCLK1 = 36MHz */
 brp = (brp / 18) / 500000;
                                      // baudrate is set to 500k bit/s
 /* set BTR register so that sample point is at about 72% bit time from bit start */
 /* TSEG1 = 12, TSEG2 = 5, SJW = 4 => 1 CAN bit = 18 TQ, sample at 72% */
                   0x03) << 24) | ((
 CAN->BTR &= ~(((
                                      0x07) << 20) | ((
                                                           0x0F) << 16)
| (
          0x1FF));
 CAN->BTR |= ((((4-1) & 0x03) << 24) | (((5-1) & 0x07) << 20) | (((12-1) & 0x0F) <<
16) | ((brp-1) & 0x1FF));
}
/*_____
                       -----
 leave initialisation mode
 *_____*/
void CAN_start (void) {
 CAN->MCR &= ~CAN_MCR_INRQ;
                                       // normal operating mode, reset INRQ
 while (CAN->MSR & CAN_MCR_INRQ);
}
/*_____
                       -----
 set the testmode
 *_____*/
void CAN_testmode (unsigned int testmode) {
 CAN->BTR &= ~(CAN_BTR_SILM | CAN_BTR_LBKM); // set testmode
```

05.5

```
Z32R 开发板
```

```
CAN->BTR |= (testmode & (CAN_BTR_SILM | CAN_BTR_LBKM));
}
/*-----
                                   -----
 check if transmit mailbox is empty
*_____*/
void CAN waitReady (void) {
 while ((CAN->TSR & CAN_TSR_TME0) == 0); // Transmit mailbox 0 is empty
 CAN_TxRdy = 1;
}
/*_____
                wite a message to CAN peripheral and transmit it
*-----*/
void CAN_wrMsg (CAN_msg *msg) {
 CAN->sTxMailBox[0].TIR = (unsigned int)0; // Reset TIR register
                                       // Setup identifier information
                                         // Standard ID
 if (msg->format == STANDARD FORMAT) {
    CAN->sTxMailBox[0].TIR |= (unsigned int)(msg->id << 21) | CAN_ID_STD;</pre>
 } else {
                                       // Extended ID
    CAN->sTxMailBox[0].TIR |= (unsigned int)(msg->id << 3) | CAN_ID_EXT;</pre>
 }
                                       // Setup type information
 if (msg->type == DATA_FRAME) {
                                         // DATA FRAME
    CAN->sTxMailBox[0].TIR |= CAN_RTR_DATA;
 }
 else {
                                       // REMOTE FRAME
    CAN->sTxMailBox[0].TIR |= CAN_RTR_REMOTE;
 }
                                       // Setup data bytes
 CAN->sTxMailBox[0].TDLR = (((unsigned int)msg->data[3] << 24) |</pre>
                      ((unsigned int)msg->data[2] << 16) |</pre>
                      ((unsigned int)msg->data[1] << 8) |</pre>
                      ((unsigned int)msg->data[0])
                                                    );
 CAN->sTxMailBox[0].TDHR = (((unsigned int)msg->data[7] << 24) |</pre>
                      ((unsigned int)msg->data[6] << 16) |</pre>
                      ((unsigned int)msg->data[5] << 8) |</pre>
                      ((unsigned int)msg->data[4]) );
                                       // Setup length
 CAN->sTxMailBox[0].TDTR &= ~CAN_TDTxR_DLC;
 CAN->sTxMailBox[0].TDTR |= (msg->len & CAN_TDTxR_DLC);
```

#### Z32R开发板

```
// enable TME interrupt
 CAN->IER |= CAN IER TMEIE;
 CAN->sTxMailBox[0].TIR |= CAN_TIxR_TXRQ; // transmit message
}
/*_____
 read a message from CAN peripheral and release it
*_____*/
void CAN rdMsg (CAN msg *msg) {
                                         // Read identifier information
 if ((CAN->sFIFOMailBox[0].RIR & CAN ID EXT) == 0) { // Standard ID
   msg->format = STANDARD_FORMAT;
  msg->id
             = (u32)0x000007FF & (CAN->sFIFOMailBox[0].RIR >> 21);
 } else {
                                             // Extended ID
   msg->format = EXTENDED_FORMAT;
           = (u32)0x0003FFFF & (CAN->sFIFOMailBox[0].RIR >> 3);
   msg->id
 }
                                         // Read type information
 if ((CAN->sFIFOMailBox[0].RIR & CAN_RTR_REMOTE) == 0) {
   msg->type = DATA_FRAME;
                                          // DATA FRAME
 } else {
   msg->type = REMOTE_FRAME;
                                          // REMOTE FRAME
 }
                                         // Read length (number of received bytes)
 msg->len = (unsigned char)0x0000000F & CAN->sFIFOMailBox[0].RDTR;
                                         // Read data bytes
 msg->data[0] = (unsigned int)0x000000FF & (CAN->sFIFOMailBox[0].RDLR);
 msg->data[1] = (unsigned int)0x000000FF & (CAN->sFIFOMailBox[0].RDLR >> 8);
 msg->data[2] = (unsigned int)0x000000FF & (CAN->sFIFOMailBox[0].RDLR >> 16);
 msg->data[3] = (unsigned int)0x000000FF & (CAN->sFIF0MailBox[0].RDLR >> 24);
 msg->data[4] = (unsigned int)0x000000FF & (CAN->sFIFOMailBox[0].RDHR);
 msg->data[5] = (unsigned int)0x000000FF & (CAN->sFIF0MailBox[0].RDHR >> 8);
 msg->data[6] = (unsigned int)0x000000FF & (CAN->sFIFOMailBox[0].RDHR >> 16);
 msg->data[7] = (unsigned int)0x000000FF & (CAN->sFIF0MailBox[0].RDHR >> 24);
 CAN->RFØR |= CAN RFØR RFOMØ;
                                          // Release FIFO 0 output mailbox
}
void CAN_wrFilter (unsigned int id, unsigned char format) {
 static unsigned short CAN_filterIdx = 0;
       unsigned int CAN_msgId
                                = 0;
```

```
if (CAN_filterIdx > 13) {
                                            // check if Filter Memory is full
   return;
 }
                                         // Setup identifier information
 if (format == STANDARD_FORMAT) {
                                            // Standard ID
     CAN_msgId |= (unsigned int)(id << 21) | CAN_ID_STD;</pre>
 } else {
                                          // Extended ID
     CAN_msgId |= (unsigned int)(id << 3) | CAN_ID_EXT;</pre>
 }
 CAN->FMR |= CAN FMR FINIT;
                                          // set Initialisation mode for filter
banks
 CAN->FAOR &= ~(unsigned int)(1 << CAN_filterIdx); // deactivate filter
                                          // initialize filter
 CAN->FSOR |= (unsigned int)(1 << CAN filterIdx);// set 32-bit scale configuration
 CAN->FMOR |= (unsigned int)(1 << CAN_filterIdx);// set 2 32-bit identifier list mode
 CAN->sFilterRegister[CAN_filterIdx].FR0 = CAN_msgId; // 32-bit identifier
 CAN->sFilterRegister[CAN_filterIdx].FR1 = CAN_msgId; // 32-bit identifier
 CAN->FFAOR &= ~(unsigned int)(1 << CAN_filterIdx); // assign filter to FIFO 0
 CAN->FAOR |= (unsigned int)(1 << CAN_filterIdx); // activate filter</pre>
 CAN->FMR &= ~CAN_FMR_FINIT;
                                           // reset Initialisation mode for filter
banks
 CAN_filterIdx += 1;
                                          // increase filter index
}
/*-----
 CAN transmit interrupt handler
 *_____
                            */
void USB_HP_CAN_TX_IRQHandler (void) {
 if (CAN->TSR & CAN_TSR_RQCP0) {
                                           // request completed mbx 0
   CAN->TSR |= CAN_TSR_RQCP0;
                                          // reset request complete mbx 0
   CAN->IER &= ~CAN_IER_TMEIE;
                                          // disable TME interrupt
   CAN_TxRdy = 1;
 }
}
```

20<u>08</u>

Z32R开发板

```
CAN receive interrupt handler
 *____
void USB_LP_CAN_RX0_IRQHandler (void) {
 if (CAN->RFOR & CAN_RFOR_FMPO) { // message pending ?
   CAN_rdMsg (&CAN_RxMsg);
                                            // read the message
                                           // set receive flag
   CAN_RxRdy = 1;
 }
}
在主程序中,我们采集 ADC 转换的结果发送出去,同时也接受发来的信息。
 for(;;) {
   i = analog[0] * 150 / 0xFFF;
     //sprintf(s, "%d", i);
     //LCD_PutString(5,80,s,Red,Yellow);
    LCD Rectangle( 5,80,5+i,86, Magenta );
    LCD_Rectangle( 6+i,80,155,86, Green );
   if (CAN_TxRdy) {
     CAN_TxRdy = 0;
     CAN_TxMsg.data[0] = analog[0]>>4; // data[0] field = ADC value
     CAN_wrMsg (&CAN_TxMsg);
                                            // transmit message
     val_Tx = CAN_TxMsg.data[0];
     }
   Delay(10000);
                                           // Wait a while to receive the message
   if (CAN_RxRdy) {
     CAN_RxRdy = 0;
     val_Rx = CAN_RxMsg.data[0];
     }
   sprintf(s, "Tx:%2X", val_Tx );
   LCD_PutString(4+52,42,s,Red,Yellow);
   sprintf(s, "Rx:%2X", val_Rx );
   LCD_PutString(4+52,60,s,Red,Yellow);
   Delay(10000);
                                          // Wait a while to receive the message
   }
```

这里我们用自检方式来做实验,我们在液晶上看到的数据是自己发的,当我们把两块相同的 STM32 板子的 CAN 总线连接到一起时需要修改 CAN 总线控制器初始化 void can\_Init (void)函

数:	
/*	*\
initialize CAN interface	1
/*	*/
<pre>void can_Init (void) {</pre>	
CAN_setup ();	<pre>// setup CAN interface</pre>
CAN_wrFilter (33, STANDARD_FORMAT);	<pre>// Enable reception of messages</pre>
/* COMMENT THE LINE BELOW TO ENABLE DEVICE	TO PARTICIPATE IN CAN NETWORK */
<pre>CAN_testmode(CAN_BTR_SILM   CAN_BTR_LBKM);</pre>	<pre>// Loopback, Silent Mode (self-test)</pre>
CAN_start ();	// leave init mode
CAN_waitReady ();	<pre>// wait til mbx is empty</pre>
}	
其中的	
<pre>CAN_testmode(CAN_BTR_SILM   CAN_BTR_LBKM);</pre>	<pre>// Loopback, Silent Mode (self-test)</pre>
要注释掉,就可以实现硬件的双机通讯了。	

Z32R 开发板

## 停十六章 定时器试验:使用中断方式

这章通过定时器1的例子来学习使用中断,程序很短,但说明问题。以下是程序的完整 代码: /\*\_\_\_\_\_\*\ | 引入相关芯片的头文件 \\*\_\_\_\_\_\*/ #include <stdio.h> #include <stm32f10x\_lib.h> // STM32F10x Library Definitions #include "STM32\_Init.h" // STM32 Initialization #include "TFT018.h" int ledLight = 0; /\*\_\_\_\_\_\*\ HARDWARE DEFINE \\*-----\*/ #define LED ( 1 << 5 ) // PB5: LED D2 /\*-----\*\ | Timer1 Update Interrupt Handler L \\*-----\*/ void TIM1\_UP\_IRQHandler (void) { if ((TIM1->SR & 0x0001) != 0) { // check interrupt source ledLight = ~ledLight; if( ledLight ) GPIOB->ODR &= ~LED; // switch on LED else GPIOB->ODR |= LED; // switch off LED // clear UIF flag TIM1->SR &=  $\sim(1<<0);$ } } // end TIM1 UP IRQHandler /\*-----\*\ MIAN ENTRY \\*\_\_\_\_\_ ---\*/ int main (void) { stm32\_Init (); // STM32 setup LCD\_Init(); LCD Clear Screen(Blue);

Font = 0;

LCD\_PutString(30,0,"STM32F 开发板",Cyan,Blue); LCD\_PutString(25,16,"定时器中断实验",Red,Yellow); Font = 1; LCD\_PutString(38,32,"Version 1.0",Green,Blue); Font = 0;LCD\_PutString(10,94,"智林测控技术研究所",Yellow,Blue); LCD\_PutString(36,111,"www.the0.net",Magenta,Blue); for(;;) { } } -----\*\ | END OF FILE ----\*/ \\*-----当然要配置一下: E:\XrEmbedded\Z32R\Software\Example\12.1 - Timer\Initialisation\SIE32\_Init.c Expand All Collapse All Help Option Value 🗄 System Timer Configuration . Real Time Clock Configuration 7 🗄 Timer Configuration - TIM1 : Timer 1 enabled ~ TIM1 period [us] 250000 TIM1 repetition counter 0 TIM1 detailed settings TIM1 interrupts TIM1\_DIER.TDE: Trigger DMA request enabled Г 7 Г TIM1\_DIER.CC4DE: Capture/Compare 4 DMA request enabled 🗌 TIM1\_DIER.CC3DE: Capture/Compare 3 DMA request enabled  $\Box$ TIM1\_DIER.CC2DE: Capture/Compare 2 DMA request enabled TIM1\_DIER.CC1DE: Capture/Compare 1 DMA request enabled Г TIM1\_DIER. UDE: Update DMA request enabled TIM1\_DIER.BIE: Break interrupt enabled TIM1\_DIER.TIE: Trigger interrupt enabled TIM1\_DIER.COMIE: COM interrupt enabled TIM1\_DIER.CC4IE: Capture/Compare 4 interrupt enabled TIM1\_DIER.CC3IE: Capture/Compare 3 interrupt enabled TIM1\_DIER.CC2IE: Capture/Compare 2 interrupt enabled TIM1\_DIER.CC1IE: Capture/Compare 1 interrupt enabled 7 TIM1\_DIER. UIE: Update interrupt enabled

Text Editor Configuration Wizard

程序运行后,我们看到发光管闪烁。

Z32R 开发板

# 第十七章 RTC 试验

## 硬件

STM32 支持电池供电的实时时钟(RTC)。



还需要一个 32.768KHz 的晶振:

			PC15	C4
				10p
	1114	32.76		C3
$\begin{array}{c} 14\\ 15\\ 16\\ 17\\ 20\\ 21\\ 22\\ 23\\ 41\\ 42\\ 43\\ 44\\ 45\\ 46\\ 49\\ 50\\ \end{array}$	PA0-WKUP/USART2_CTS/ADC_IN0/TIM2_CH1_ETR PA1/USART2_RTS/ADC_IN1/TIM2_CH2 PA2/USART2_TX/ADC_IN2/TIM2_CH3 PA3/USART2_RX/ADC_IN3/TIM2_CH4 PA4/SPI1_NSS/USART2_CK/ADC_IN4 PA5/SPI1_NSS/USART2_CK/ADC_IN4 PA5/SPI1_SCK/ADC_IN5 PA6/SPI1_MIS0/ADC_IN6/TIM3_CH1 PA7/SPI1_MOSI/ADC_IN7/TIM3_CH2 PA8/USART1_CK/TIM1_CH1/MCO PA9/USART1_CK/TIM1_CH2 PA10/USART1_CTS/CANRX / USBDM (2)/TIM1_CH4 PA12 / USART1_RTS/CANTX / USBDP (2)/TIM1_ETR PA13/JTMS-SWDAT PA14/JTCK-SWCLK PA15/JTDI	PC15-OSC32_OUT PC14-OSC32_IN PC13-ANTI_TAMP PC12 PC11 PC10 PC10 PC9 PC8 PC7 PC6 PC5/ADC_IN15 PC4/ADC_IN14 PC3/ADC_IN13 PC2/ADC_IN12 PC1/ADC_IN11	4 3 PC14 2 53 52 51 40 39 38 37 25 24 11 10 9 8	<b>11</b> - <u>1</u> -3*
26 27 28 55	PB0/ADC_IN8/TIM3_CH3 PB1/ADC_IN9/TIM3_CH4 PB2 / BOOT1 PB3/JTD0	OSC_IN ·	5	
56 57 58 59 61 62 29 30 33 34 35 36	PB3/JTRST PB4/JTRST PB5/I2C1_SMBAI PB6/I2C1_SCL/TIM4_CH1 PB7/I2C1_SDA/TIM4_CH2 PB8/TIM4_CH3 PB9/TIM4_CH4 PB10/I2C2_SCL/USART3_TX PB11/I2C2_SDA/USART3_TX PB12/SPI2_NSS/I2C2_SMBA/USART3_CK/TIM1_BKIN PB13/SPI2_SCK/USART3_CTS/TIM1_CH1N PB14/SPI2_MISO/USART3_RTS/TIM1_CH2N PB15/SPI2_MOSI/TIM1_CH3N	OSC_OUT PD2 NRST BOOT0	6 54 7 60	
	STM32F103R			

\*\

软件

/\*

HARDWARE DEFINE

```
Z32R 开发板
                                    智林测控技术研究所 www.the0.net
\*----
                                                         --*/
#define LED
                       (1 << 5)
                                                 // PB5: LED D2
                                                   -----*\
/*-----
RTC Interrupt Handler
\*-----
                                                           -*/
void RTC_IRQHandler(void) {
 if (RTC->CRL & (1<<0) ) {
                                                 // check second flag
   RTC->CRL &= (1 << 0);
                                                 // clear second flag
   ledLight = ~ledLight;
   if( ledLight )
       GPIOB->ODR &= ^{\sim}LED;
                                                    // switch on LED
   else
       GPIOB \rightarrow ODR = LED;
                                                     // switch off LED
   }
 if (RTC->CRL & (1<<1) ) {
                                                // check alarm flag
   RTC->CRL &= (1 << 1);
                                                // clear alarm flag
   Alarm = 1;
   }
 } // end TIM1_UP_IRQHandler
/*-----
                                                           -*\
RTC Interrupt Handler
\*-----
                                                           */
void DisplayTime(void) {
 unsigned int TimeVar=0, THH = 0, TMM = 0, TSS = 0;
 char s[30];
 TimeVar = RTC->CNTH << 16 | RTC->CNTL;
 /* Compute hours */
 THH = TimeVar/3600;
 /* Compute minutes */
 TMM = (TimeVar % 3600)/60;
 /* Compute seconds */
 TSS = (TimeVar % 3600)% 60;
 sprintf(s, "%2d 时%2d 分%2d 秒", THH, TMM, TSS);
 LCD PutString (30, 50, s, Red, Yellow);
 }
/*-----
                                                           *\
MIAN ENTRY
\*-----
                                                           */
int main (void) {
 stm32_Init ();
                                               // STM32 setup
 LCD_Init();
```

#### 智林测控技术研究所 www.the0.net

\*\

-\*/

```
LCD_Clear_Screen(Blue);
```

END OF FILE

Z32R开发板

\\*-----\* 这段程序在液晶屏幕上显示当前时间。同时中断程序使发光管闪烁。 首先,初始化 RTC,接着设置一个时间,然后读取时钟数据并且显示。

## 第十八章 外部中断试验

硬件



我们板上有两个按钮 PB2 和 PB3,这个实验我们用这两个按钮产生外部中断。

软件



```
Z32R开发板
                                            智林测控技术研究所 www.the0.net
05.5
        1*-
                                                                   -*\
        HARDWARE DEFINE
                                                                  --*/
        \*-----
                     (1 << 5)
       #define LED
                                                         // PB5: LED D2
       /*-----
                                                           -----*\
        EXTIO Interrupt Handler
        \*-----
                                                                   -*/
       void EXTI0_IRQHandler(void)
        {
         if (EXTI->PR & (1<<0)) {
                                                         // EXTI13 interrupt pending?
           if ((ledLight ^=1) == 0)
               GPIOB->ODR &= ~LED;
                                                             // switch on LED
           else
               GPIOB \rightarrow ODR \mid = LED;
                                                             // switch off LED
           EXTI -> PR \mid = (1 << 0);
                                                          // clear pending interrupt
         }
       }
        /*----
                                                                    -*\
        EXTI15..10 Interrupt Handler
        \*-----
                                                                    */
       void EXTI15 10 IRQHandler(void)
        {
         if (EXTI->PR & (1<<13)) {
                                                         // EXTIO interrupt pending?
           if ((ledLight ^=1) == 0)
               GPIOB->ODR &= ^{\sim}LED;
                                                             // switch on LED
           else
               GPIOB \rightarrow ODR = LED;
                                                             // switch off LED
           EXTI->PR |= (1 << 13);
                                                          // clear pending interrupt
         }
        }
        /*-----
                                                                    *\
        MIAN ENTRY
       \*-----
       int main (void) {
         stm32_Init ();
                                                       // STM32 setup
         LCD_Init();
         LCD_Clear_Screen(Blue);
         Font = 0;
         LCD_PutString(30,0, "STM32F 开发板", Cyan, Blue);
         LCD_PutString(35, 16, "外部中断实验", Red, Yellow);
```

```
2008
```

```
Font = 1;
LCD_PutString(38,32, "Version 1.0", Green, Blue);
Font = 0;
LCD_PutString(10,94, "智林测控技术研究所", Yellow, Blue);
LCD_PutString(36,111, "www.the0.net", Magenta, Blue);
for(;;) {
}
}
/*------*\
| END OF FILE |
/*------*/
```

软件运行后,按按钮 PB2 和 PB3,都会导致中断发生,反转发光管。

## 第十九章 入侵检测试验

#### 硬件



按钮 PB2 接在侵入检测管脚上。

软件



```
| TAMPER Interrupt Handler
                                                    Т
\*_____*/
void TAMPER_IRQHandler(void)
{
 if (BKP->CSR & (1<<9) ) {
                                          // Tamper interrupt flag
   PWR->CR |= (1<<8);
                                          // enable access to RTC,
BDC registers
   BKP->CSR |= (1<<1);
                                          // clear Tamper Interrupt
   BKP - >CSR \mid = (1 < < 0);
                                          // clear tamper Event
                                          // disable access to RTC,
   PWR->CR &= ~(1<<8);
BDC registers
   if ((BKP->DR1 == 0) &&
      (BKP \rightarrow DR2 == 0))
    GPIOB->ODR |= LED;
                                           // switch off LED
   else
                                           // switch on LED
    GPIOB->ODR &= ~LED;
   }
}
/*-----*\
 MIAN ENTRY
\*-----
int main (void) {
 stm32_Init ();
                                        // STM32 setup
 PWR->CR |= (1<<8);
                                          // enable access to RTC,
BDC registers
                                          // fill BKP_DR1 register
 BKP \rightarrow DR1 = 0 \times 55 AA;
 BKP \rightarrow DR2 = 0x33CC;
                                          // fill BKP_DR2 register
 PWR -> CR \&= \sim (1 << 8);
                                          // disable access to RTC,
BDC registers
 LCD_Init();
 LCD_Clear_Screen(Blue);
 Font = 0;
 LCD PutString(30,0,"STM32F 开发板",Cyan,Blue);
 LCD_PutString(35,16,"侵入检测实验",Red,Yellow);
 Font = 1;
 LCD_PutString(38,32,"Version 1.0",Green,Blue);
 Font = 0;
 LCD PutString(10,94,"智林测控技术研究所",Yellow,Blue);
 LCD_PutString(36,111,"www.the0.net",Magenta,Blue);
```



### 当然,需要配置:

E:\XrEmbedded\Z32R\Software\Exam	ple\15.1 - Tamper\Initialisation\STM32_Init.c	
Expand All Collapse All	Help	
Option	Value	
🕀 Clock Configuration		
🕂 Independent Watchdog Configuration		
🕒 System Timer Configuration		
🕂 Real Time Clock Configuration		
🛨 Timer Configuration		
🕀 - USART Configuration		
External interrupt/event Configuration		
Tamper Configuration		
- Tamper Pin enable	$\checkmark$	
Tamper pin active level	active level = LOW	
Tamper interrupt enable		
+ Embedded Flash Configuration		
,		
Text Editor Configuration Wizard		
2008

# 第二十章 看门狗试验

软件

```
/*-----*\
| 引入相关芯片的头文件
                                \*-----*/
#include <stdio.h>
#include <stm32f10x_lib.h> // STM32F10x Library Definitions
#include "STM32_Init.h" // STM32 Initialization
#include "TFT018.h"
int ledLight = 0;
/*-----*\
HARDWARE DEFINE
                                 \*-----*/
#define LED (1 << 5)</pre>
                           // PB5: LED D2
/*-----*\
                                 Delay
| 延时 Inserts a delay time.
                                 T
| nCount: 延时时间
nCount: specifies the delay time length.
\*_____*/
void Delay(unsigned int nCount) {
 for(; nCount != 0; nCount--);
 }
/*-----*\
| MIAN ENTRY
                                 \*-----*/
int main (void) {
 int i;
 stm32_Init ();
                         // STM32 setup
 LCD Init();
 LCD_Clear_Screen(Blue);
 Font = 0;
 LCD_PutString(30,0,"STM32F 开发板",Cyan,Blue);
 LCD_PutString(25,16,"看门狗复位实验",Red,Yellow);
 Font = 1;
```

#### 05.5

```
Z32R 开发板
                            智林测控技术研究所 www.the0.net
 LCD_PutString(38,32,"Version 1.0",Green,Blue);
 Font = 0;
 LCD_PutString(10,94,"智林测控技术研究所",Yellow,Blue);
 LCD_PutString(36,111,"www.the0.net",Magenta,Blue);
 if (RCC->CSR & (1<<29)) {
                                        // IWDG Reset Flag set
   RCC->CSR |= (1<<24);
                                        // Clear Reset Flags
                                       // switch on LED
  GPIOB->ODR |= LED;
   LCD_PutString(45,50,"看门狗复位",Yellow,Red);
  }
 else {
  GPIOB->ODR &= ~LED;
                                        // switch off LED
   LCD_PutString(45,50, "开机复位 ",Blue,Yellow);
   }
 for (i = 0; i < 10; i++) {
  Delay(1000000);
                                       // wait less than watchdog
interval
   IWDG->KR = 0xAAAA;
                                       // reload the watchdog
 }
 GPIOB->ODR |= LED;
                                      // switch on LED
 for(;;) {
  }
 }
/*_____
                                              ----*\
 | END OF FILE
\*-----*/
```

#### 配置:

E:\XrEmbedded\Z32R\Software\Exa	ple\16.1 - IWDG\Initialisation\SIM32_Init.c				
Expand All Collapse All	Help				
Option	Value				
🕀 Clock Configuration					
🖃 Independent Watchdog Configuration	✓				
IWDG period [us]	2000000				
庄 System Timer Configuration					
🕀 Real Time Clock Configuration					
🛨 Timer Configuration					
USART Configuration					
External interrupt/event Configuration					
🕂 Tamper Configuration					
🗄 General purpose I/O Configuration	V				
🗄 Embedded Flash Configuration	V				
\ Text Editor \ Configuration Wizard /					

程序运行后,会看到硬件复位信息,当程序停止喂狗后,会显示出看门狗复位信息。

2008

# 第二十一章 软件中断试验

## 软件

先看看软件中断	所处理函	数:		
asm void SVC	_Handle	r (void) {		
PRESERVE8				
	TST	LR,#4	; Called from Handler Mode?	
	MRSNE	R12,PSP	; Yes, use PSP	
	MOVEQ	R12,SP	; No, use MSP	
	LDR	R12,[R12,#24]	; Read Saved PC from Stack	
	LDRH	R12,[R12,#-2]	; Load Halfword	
	BICS	R12,R12,#0xFF00	; Extract SVC Number	
	PUSH	{R4,LR}	; Save Registers	
	LDR	LR,=SVC_Count		
	LDR	LR,[LR]		
	CMP	R12,LR		
	BHS	SVC_Dead	; Overflow	
	LDR	LR,=SVC_Table		
	LDR	R12,[LR,R12,LSL #2]	; Load SVC Function Address	
	BLX	R12	; Call SVC Function	
	POP	{R4,LR}		
	TST	LR,#4		
	MRSNE	R12,PSP		
	MOVEQ	R12,SP		
	STM	R12,{R0-R3}	; Function return values	
	BX	LR	; RETI	
SVC Dead				
_	В	SVC_Dead	; None Existing SVC	
SVC Cnt	EQU	(SVC End-SVC Table)/4		
_ SVC_Count	DCD	SVC_Cnt		
: Import user	SVC fun	ctions here.		
,	IMPORT	SVC Ø		
	IMPORT	SVC 1		
	IMPORT	SVC 2		
	IMPORT	SVC 3		

```
Z32R开发板
                                        智林测控技术研究所 www.the0.net
05.5
       SVC_Table
       ; Insert user SVC functions here
                   DCD
                          ___SVC_0
                                             ; SVC 0 Function Entry
                   DCD
                          ___SVC_1
                                             ; SVC 1 Function Entry
                   DCD
                          ___SVC_2
                                             ; SVC 2 Function Entry
                          SVC 3
                   DCD
                                             ; SVC 2 Function Entry
       SVC_End
                   ALIGN
       }
       /*____
                                         * end of file
       *_____*/
       软件中断后,按照软件中断号分别执行__SVC_0、__SVC_1、__SVC_2、__SVC_3。这四个函数定义
       如下:
       int __svc(0) add (int i1, int i2);
       int ___SVC_0
                   (int i1, int i2) {
        return (i1 + i2);
       }
       int __svc(1) mul4(int i);
       int SVC 1
                   (int i) {
        return (i << 2);
       }
       int __svc(2) div (int i1, int i2);
       int SVC 2
                     (int i1, int i2) {
        return (i1 / i2);
       }
       int __svc(3) mod (int i1, int i2);
       int __SVC_3 (int i1, int i2) {
        return (i1 % i2);
       }
       在主函数里测试:
       int res;
       char s[30];
       /*-----
       _ _ _ _ _ _ _ _
        Test Function
```

2008

```
*_____
----*/
void test_t (void) {
 res = div (res, 10);
                                       // Call SWI Functions
 sprintf(s,"div:%d",res);
 LCD_PutString(10,80,s,Magenta,Blue);
 res = mod (res, 3);
 sprintf(s,"mod:%d",res);
 LCD_PutString(90,80,s,Magenta,Blue);
}
/*____
                            ------
 Test Function
----*/
void test_a (void) {
 res = add (74, 27);
                                        // Call SWI Functions
 sprintf(s,"add:%d",res);
 LCD_PutString(10,60,s,Magenta,Blue);
 res += mul4(res);
 sprintf(s,"mul4:%d",res);
 LCD_PutString(90,60,s,Magenta,Blue);
}
/*-----*\
 MIAN ENTRY
\*_____
                                                   --*/
int main (void) {
 stm32_Init ();
                                       // STM32 setup
 LCD_Init();
 LCD_Clear_Screen(Blue);
 Font = 0;
 LCD_PutString(30,0,"STM32F 开发板",Cyan,Blue);
 LCD_PutString(30,16,"软件中断实验",Red,Yellow);
 Font = 1;
 LCD_PutString(38,32,"Version 1.0",Green,Blue);
 Font = 0;
```

### 05.5

#### Z32R开发板

```
LCD_PutString(10,94,"智林测控技术研究所",Yellow,Blue);
LCD_PutString(36,111,"www.the0.net",Magenta,Blue);
sprintf(s,"res:%d",res);
LCD_PutString(20,40,s,Magenta,Blue);
test_a();
test_t();
for(;;) {
}
}
/*------*\
END OF FILE |
*------*/
```

## 第二十二章 实时操作系统试验:uC/OS-II

首先,版权问题。

### uC/OS-II 作者致中国用户的公开信

### 致 Micrium 产品的使用者:

Micrium 公司产品包括  $\mu$ C/OS-II,  $\mu$ C/GUI,  $\mu$ C/FS,  $\mu$ C/TCP-IP,  $\mu$ C/USB 等。 Micrium 公 司提供嵌入式系统应用方面的产品,并对其软件拥有知识产权。 我公司花费了大量的时间 和财力为嵌入式领域提供高质量的软件产品。我们的所有产品都以源代码的形式提供给客户, 具有极大的适用性。我们的产品不是免费软件,也不是开放源码的软件,因此,不能免费使 用。 您可以通过购买我关于  $\mu$ C/OS-II 的书而得到  $\mu$ C/OS-II 源代码,当您从芯片厂商那里 购买评估板时,您也可以获得  $\mu$ C/OS-II 源代码,但您必须同时购买一本含有 CD 的我的书。 当您用于商业目的,您必须购买使用授权,这在书中有明确规定。 我公司其它软件如  $\mu$ C/GUI,  $\mu$ C/FS,  $\mu$ C/TCP-IP,  $\mu$ C/USB 等的销售模式与  $\mu$ C/OS-II 不同,如果您没有购买 使用授权,您完全不可以拥有该源代码,也不能将源代码用于产品。因此,请不要非法使用和 发布我们的软件。如果您已经有了  $\mu$ C/OS-II 的代码,但您并没有我的关于  $\mu$ C/OS-II 的书,请您购买一本。 如果您正在将  $\mu$ C/OS-II 用于您的产品,您需要购买并获得正式使用授权。如果您有一份我公司其它产品的拷贝,您必须销毁非法拷贝并购买使用授权。 Micrium 公 司系列产品和使用授权由北京麦克泰软件技术有限公司在中国独家代理。 感谢您的诚实合作和理解!

只是一个提醒,如果大家在量产产品中使用,被抓到罚款,可就不划算了。

#### 我们对操作系统使用的看法

我的意见是,很多场合 RTOS 是可以不用的,有时候用了反倒麻烦。像 NXP 系列的 ARM 我 个人称其为"小 ARM",就是片上带有 Flash 的这种,我们可以把这样 ARM 当作一个功能强 大的单片机来用。当然,我不是反对使用 RTOS,要看具体项目来定,需要用就用。

#### 软件

这是个简单的例子,建立 2 个任务,运行后一个任务使 LED 闪烁,另一个在液晶上显示任务 切换数。源码在光盘上。