



STM32F0系列 **ARM Cortex-M0核** **微控制器**开发与应用

喻金钱 喻斌 袁芳 编著

ARM Cortex-M0 ARM Cortex-M0 ARM Cortex-M0 ARM Cortex-M0 ARM Cortex-M0



清华大学出版社

全 部 容 容

STM32F0 系列 ARM Cortex-M0 核微控制器开发与应用

喻金钱 喻 斌 袁 芳 编著

清华大学出版社
北 京

内 容 简 介

本书从实际应用需求和开发过程中所遇到的问题出发,介绍了STM32F0系列ARM芯片内外设和各个功能模块的应用。

本书没有详解一些有关芯片的储存结构系统构架、指令集等理论性的知识,而是从最基本的应用需求出发,结合大量实例,依托库函数,详细讲解了I/O口、异步串口、节拍定时器、SPI接口、RTC、看门狗、定时器、I2C接口、DMA接口、模数转换器等外设接口的使用方法。本书注重实际操作和开发中的细节,讲解过程循序渐进,对在开发过程中容易出现的错误情况作出提醒,并与读者分享作者在实际开发中的一些经验和感想。为有单片机和C语言基础的读者,打开了通向嵌入式世界的大门。

本书可以作为单片机爱好者的学习用书,也可以作为嵌入式应用工程技术人员的学习和培训用书,同时还可以作为大学生学习单片机的教材。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

STM32F0 系列 ARM Cortex-M0 核微控制器开发与应用/喻金钱,喻斌,袁芳编著. —北京:清华大学出版社,2015

ISBN 978-7-302-37431-2

I. ①S… II. ①喻… ②喻… ③袁… III. ①微控制器 IV. ①TP332.3

中国版本图书馆 CIP 数据核字(2014)第 170786 号

责任编辑:钟志芳

封面设计:刘超

版式设计:文森时代

责任校对:王云

责任印制:杨艳

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社总机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印装者:清华大学印刷厂

经 销:全国新华书店

开 本:185mm×260mm

印 张:26.75

字 数:631 千字

(附光盘 1 张)

版 次:2015 年 1 月第 1 版

印 次:2015 年 1 月第 1 次印刷

印 数:1~4000

定 价:52.00 元

前 言

ARM 是嵌入式微处理器行业的最卓越供应商,这已成为不争的事实。ARM 公司推出的各种各样基于通用架构的处理器具有高性能和行业领先的功效,使系统成本大幅度降低。迄今为止,ARM 公司有超过 900 个可提供硅、工具和软件的合作伙伴,生产出超过 250 亿个处理器,每天的销售超过 1600 万。

嵌入系统的 ARM 化趋势越来越明显,在智能化手持终端设备中,ARM 通过合作厂商占据绝大部分市场,在传统的微控制器领域,ARM 推出有针对性的系列,一步一步挤占其他内核的市场份额。

在群雄盘踞的嵌入式市场中,ARM 是如何突围的呢?凭借提供众多外设功能、卓越的性能、更低的功耗和更小的费用,ARM 傲视群雄。在相同的功能和性能下,ARM 消耗更少的能量,支出更低廉的费用;在同样的能量限制下,提供更快的速度;在同样的预算下,达到更高的性能,这就是 ARM 的秘诀。

单片机市场的规模越来越大,到 2012 年,估计出货量在 25G 片左右,世界各大半导体公司纷纷亮出各具特色的器件和架构。ARM 公司也不示弱,推出针对 8 位和 16 位机市场的 ARM Cortex-M 内核。ARM Cortex-M4 内核针对高端市场,需要高速运算应用,挤占一部分先前需要使用 DSP 的市场;ARM Cortex-M3 内核针对中端市场,凭借优异的性能和众多的外设以及更低的价格来抢占这部分市场;ARM Cortex-M0 内核,专门针对先前 8 位机市场,在相同价格上,提供更高性能和更低功耗。

目前已有众多 MCU 厂家推出了带有 ARM 内核产品系列,不同厂家依据自身特点整合了功能各异的外设,形成了丰富的 ARM 产品组态,给用户更多的选择空间。

纵观国内高校嵌入式教学,几年前,是 51 内核教材一统天下的局面。技术进步和市场规模的变化,使得高校嵌入式教学内容远远背离了现实的需要,面对众多的内核,让高校教学也陷入了困惑中。这几年 ARM 的发展和应用的普及,让高校嵌入式教学找到转型的方向,许多高校已经开始选用 ARM 作为嵌入式教学内容,这种趋势越来越明显。

本书主要内容

本书在编写过程中,强调实用、易用和有用,共分为 3 大部分:第一部分(第 1~3 章),介绍开发板硬件结构和软件开发平台,是后面学习的基础;第二部分(第 4~15 章),介绍 STM32F0 芯片的各个功能模块的特点、库函数,然后用具体的实例详细介绍如何来用这些库函数来实现功能模块的不同应用;第三部分(第 16~18 章),基于 STM32F0 常用系统设计的实例应用。

第一部分是本书的基础,需要熟练掌握。这样才可以有效提高开发效率,减少错误的发生。

第二部分是本书的重点。用 13 章的篇幅分别讲述各个功能模块。每章采用了相同的讲

解模式，首先介绍该模块的功能，再介绍能实现这些功能的库函数，最后用多个实例详细讲解如何使用这些库函数实现模块功能。这部分内容是本书最有特色的，也是读者最感兴趣的，主要有 LED 灯驱动实例、流水灯实例、按键实例、串口接收数据实例、串口发送数据实例、串口队列实例、嘀嗒时钟、实用按键实例、外部中断实例、中断嵌套实例、万年历实例、I2C 与 24C02 通信实例、单通道 ADC 采样实例、温度采样实例、内部 FLASH 读写实例、定时器简单定时实例、不同占空比 PWM 输出实例、不同频率不同占空比 PWM 输出实例、独立看门狗实例、窗口看门狗实例、DAC 实例等 20 多个实例。

第三部分是本书的提高部分，讲述了 MCU 的高级应用和常用的系统设计的实例。这些实例在实际中经常使用，本书对这些实例不是泛泛而谈，而是详尽讲解其思路和逻辑方法，这也是本书的特色。这一部分内容包括 USART 串口和 SPI 接口 DMA 接收和发射、使用 DMA 采集多通道 ADC、NOR FLASH 数据储存方案、LCD 屏等高级应用。

本书并不去介绍每个寄存器的详细功能和具体使用（这在 STM32F0 的技术手册中有详细的描述），而是通过对具体实例的讲解和剖析，结合厂家提供的固件库，让读者能很容易、很简单地使用 STM32F0 系列芯片，并能把这一系列芯片的功能发挥到极致。只要会 C 语言，通过本书的学习，读者就能使用 ARM，不必去弄懂底层硬件结构，就可以很好地使用 STM32F0 系列来进行开发调试工作。

通过使用固件函数库，无需深入掌握细节，用户也可以轻松应用每一个外设。因此，使用固态函数库可以大大减少用户的程序编写时间，进而降低开发成本。

本书以具体的功能实例为基础，引导读者分析实例并实现这些功能。即在开发调试中，如何去一步一步地解决问题和实现功能，以及怎样把一个复杂的问题划分成一个个好解决的小问题来实现整个功能。本书着重介绍解决问题的方法。

读者对象

本书要求读者有基本的 C 语言知识，不需要有硬件方面的知识。通过本书的学习，可以使初学者很快进入到嵌入式开发的大门。

如果读者是个单片机方面的高手，想通过本书学习使用 STM32F0 系列的芯片，只需要熟悉开发平台和实验平台，然后了解每个外设功能模块是如何初始化即可。第三部分内容可以看看，或许能有意想不到的收获。

如果读者是一个单片机的入门者，需要按照本书的编排顺序，一章一章往后学习，一个一个例程去理解和编写，就像堆积木一样，进行功能堆积。学完本书，不说是高手，也是一个嵌入式开发的熟手。

同时，本书可作为嵌入式应用工程技术人员的学习和培训用书，也可以作为企业内部培训教材，如果作为大学单片机课程的教材，将会起到事半功倍的教学效果。

本书配套

本书配套光盘中有书中各个实例的源代码，这些源代码在实验板上已验证通过。希望广大读者不要只是把源代码一烧了之，而是应该尝试着自己亲自编写这些软件。只有经过

不断地实践，才能获得真知。

本书配套实验板可帮助广大读者能很快地学习嵌入式开发。读者也可根据本书提供的原理图自行搭建，或使用其他实验板，依据其硬件更改相应实例代码即可。作者在此提供硬件支持，是让读者能在开始时绕过硬件屏障，提供一个经过验证的可靠的硬件平台，让读者全心学习 STM32F0 系列芯片功能外设。当读者掌握了这些技能后，依据本书中作者提供的原理图内容和注意细节，完全可以设计出实现自己所需功能的、性能优异的电路板。读者如需购买本开发板，可以与作者联系购买事宜。

整个开发系统的搭建，需要一台 PC 机、一个实验板、一个 J-LINK 调试器、一根串口延长线即可。

本书在编写过程中，得到了家人的理解和大力支持，得到了清华大学出版社编辑老师的大力支持，在此表示感谢。本书由喻金钱、喻斌、袁芳编著，程基峰、胡振等人也参与了本书的编写和例程调试及测试工作。

由于本书涉及的知识领域日新月异，加之作者水平有限及时间仓促，书中难免有疏漏和不足之处，在此诚挚希望读者批评指正！如有任何疑虑或批评指正，可以和作者联系（tonda@126.com），以便我们改进与提高，特此感谢。

喻金钱

目 录

第 1 章 STM32F051xx 系列芯片简介 ... 1

1.1 STM32F051xx 系列芯片功能

简介 1

1.2 功能概述 3

1.2.1 内核 3

1.2.2 存储器 3

1.2.3 循环冗余校验计算单元 (CRC) 3

1.2.4 直接存储器访问控制器 (DMA) 3

1.2.5 向量嵌套中断控制器 (NVIC) 4

1.2.6 扩展中断/事件控制器 (EXTI) 4

1.2.7 时钟和启动 4

1.2.8 引导模式 4

1.2.9 电源管理 5

1.2.10 低功耗模式 5

1.2.11 实时时钟 (RTC) 和后备寄存器 6

1.2.12 定时器 6

1.2.13 看门狗 8

1.2.14 SysTick 定时器 8

1.2.15 两线串行接口 I2C 8

1.2.16 通用同步/异步收发器 (USART) 9

1.2.17 高清晰度多媒体接口 (HDMI) ——

消费电子控制 (CEC) 9

1.2.18 通用输入/输出端口 (GPIO) 9

1.2.19 触摸传感控制器 (TSC) 9

1.2.20 模数转换器 (ADC) 9

1.2.21 数模转换器 (DAC) 10

1.2.22 快速比较器 10

1.2.23 两线串行调试端口 (SW-DP) 10

第 2 章 开发板硬件结构 11

2.1 电路原理图 11

2.2 原理图说明 13

2.2.1 电源电路 13

2.2.2 系统复位电路 14

2.2.3 时钟电路 14

2.2.4 JTAG 接口电路 15

2.2.5 串口电路 15

2.2.6 键盘电路 15

2.2.7 LED 灯电路 16

2.2.8 I2C 接口电路 17

2.2.9 ADC 电路 18

2.2.10 SPI 接口电路 18

2.3 开发板元器件布局图 19

第 3 章 编译开发环境的建立 21

3.1 下载和安装 EWARM 21

3.2 IDE 界面简介 23

3.3 生成一个新项目 24

3.3.1 建立项目文件目录并复制文件 24

3.3.2 生成新的工作区 25

3.3.3 生成新项目 25

3.3.4 给项目添加文件 27

3.4 配置项目选项 28

3.4.1 通用选项设置 (General Options) 28

3.4.2 C/C++ 编译器选项设置 (C/C++

Compiler) 29

3.4.3 Assembler 选项设置 (Assembler) 30

3.4.4 Output Converter 选项设置 30

3.4.5 Debugger 选项设置 31

3.5 串口调试助手介绍 31

第 4 章 通用和复用 I/O 口 33

4.1 GPIO 功能描述 33

4.1.1 GPIO 主要特性 33

4.1.2 GPIO 主要功能 33

4.1.3 通用 I/O 口 34

4.1.4 I/O 引脚的复用功能和重映射 34

4.1.5 I/O 端口控制寄存器 35

4.1.6 I/O 端口数据寄存器 35

4.1.7 I/O 数据位处理	35	4.6 I/O 口输入实例 1——按键输入 1 ...	64
4.1.8 GPIO 口锁定	36	4.6.1 实例要求	64
4.1.9 I/O 口复用功能	36	4.6.2 硬件基础	64
4.1.10 外部中断和唤醒	36	4.6.3 软件结构	64
4.1.11 输入配置	36	4.6.4 实例代码	65
4.1.12 输出配置	36	4.6.5 编译下载和调试	67
4.1.13 模拟输入配置	37	4.7 I/O 口输入实例 2——按键输入 2 ...	69
4.2 GPIO 库函数	37	4.7.1 实例要求	69
4.2.1 函数 GPIO_DeInit	38	4.7.2 硬件基础	69
4.2.2 函数 GPIO_Init	38	4.7.3 软件结构	69
4.2.3 函数 GPIO_PinLockConfig	39	4.7.4 实例代码	70
4.2.4 函数 GPIO_ReadInputDataBit	40	第 5 章 USART 串口的一般应用	73
4.2.5 函数 GPIO_ReadInputData	40	5.1 USART 功能描述	73
4.2.6 函数 GPIO_ReadOutputDataBit	41	5.1.1 USART 主要功能	73
4.2.7 函数 GPIO_ReadOutputData	41	5.1.2 USART 扩展功能	74
4.2.8 函数 GPIO_SetBits	41	5.1.3 USART 不同工作模式	74
4.2.9 函数 GPIO_ResetBits	42	5.1.4 发送器	75
4.2.10 函数 GPIO_WriteBit	42	5.1.5 接收器	76
4.2.11 函数 GPIO_Write	43	5.1.6 分数比特率的产生	77
4.2.12 函数 GPIO_PinAFConfig	43	5.1.7 自动比特率检测	77
4.3 位运算	45	5.1.8 多机通信	78
4.3.1 移位运算	45	5.1.9 USART 同步模式	79
4.3.2 按位“与”运算(&)	47	5.1.10 单线半双工模式	80
4.3.3 按位“或”运算()	48	5.1.11 智能卡模式	81
4.3.4 取反运算(~)	49	5.1.12 IrDA SIR ENDEC 功能模块	82
4.3.5 异或运算符(^)	49	5.1.13 用 DMA 实现连续通信	84
4.4 I/O 口输出实例 1——控制 LED 灯	49	5.1.14 USART 中断	85
4.4.1 实例目的和要求	49	5.2 USART 串口库函数	85
4.4.2 硬件基础	50	5.2.1 函数 USART_Init	85
4.4.3 软件结构	50	5.2.2 函数 USART_Cmd	87
4.4.4 实例代码	50	5.2.3 函数 USART_ITConfig	88
4.4.5 编译下载	59	5.2.4 函数 USART_SendData	89
4.5 I/O 口输出实例 2——流水灯	60	5.2.5 函数 USART_ReceiveData	89
4.5.1 实例要求	60	5.2.6 函数 USART_GetFlagStatus	89
4.5.2 硬件基础	60	5.2.7 函数 USART_ClearFlag	90
4.5.3 软件结构	61	5.2.8 函数 USART_GetITStatus	91
4.5.4 实例代码	61	5.2.9 函数 USART_ClearITPendingBit	92
4.5.5 编译下载	63	5.3 USART 通信实例 1——串口发送数据	92

5.3.1 实例要求	92	7.2.7 函数 EXTI_GetITStatus	122
5.3.2 硬件基础	92	7.3 外部中断实例	122
5.3.3 软件结构	93	7.3.1 实例目的	122
5.3.4 实例代码	95	7.3.2 实例要求	122
5.3.5 编译下载和调试	97	7.3.3 硬件基础	123
5.4 USART 通信实例 2——中断接收数据	98	7.3.4 软件结构	123
5.4.1 实例要求	98	7.3.5 实例代码	125
5.4.2 硬件基础	98	7.3.6 编译下载和调试	127
5.4.3 软件结构	99	7.4 中断嵌套实例	127
5.4.4 实例代码	99	7.4.1 实例目的	127
5.4.5 编译下载和调试	101	7.4.2 实例要求	127
5.5 使用队列收发数据实例	101	7.4.3 硬件基础	127
第 6 章 系统定时器	104	7.4.4 软件结构	127
6.1 系统定时器概述	104	7.4.5 实例代码	127
6.2 库函数介绍	104	7.4.6 编译下载和调试	130
6.3 系统定时器实例——节拍定时器实例	105	第 8 章 实时时钟 (RTC)	131
6.3.1 实例要求	105	8.1 实时时钟简介	131
6.3.2 软件结构	105	8.1.1 RTC 主要特性	131
6.3.3 实例代码	105	8.1.2 RTC 功能模块	131
6.3.4 编译下载和调试	106	8.1.3 时钟和预分频器	132
6.4 有实际应用意义的键盘实例	106	8.1.4 实时时钟和日历	133
6.4.1 实例要求	106	8.1.5 可编程报警	133
6.4.2 软件结构	107	8.1.6 RTC 初始化及配置	133
6.4.3 实例代码	107	8.1.7 读日历寄存器	135
6.4.4 编译下载和调试	109	8.1.8 复位过程	136
第 7 章 中断和事件	113	8.1.9 RTC 同步	136
7.1 中断和事件概述	113	8.1.10 RTC 参考时钟检测	136
7.1.1 嵌套向量中断控制器 (NVIC)	113	8.1.11 RTC 平滑数字校准	137
7.1.2 外部中断和事件控制器 (EXTI)	114	8.1.12 时间戳功能	138
7.2 库函数介绍	117	8.1.13 侵入检测	139
7.2.1 函数 NVIC_Init	117	8.1.14 校准时钟输出	140
7.2.2 函数 EXTI_DeInit	119	8.1.15 报警输出	141
7.2.3 函数 EXTI_Init	119	8.1.16 RTC 低功耗模式	141
7.2.4 函数 EXTI_GenerateSWInterrupt	121	8.1.17 RTC 中断	141
7.2.5 函数 EXTI_GetFlagStatus	121	8.2 RTC 实时时钟库函数介绍	142
7.2.6 函数 EXTI_ClearFlag	121	8.2.1 函数 RTC_Init	142
		8.2.2 函数 RTC_SetTime	143
		8.2.3 函数 RTC_GetTime	144
		8.2.4 函数 RTC_GetSubSecond	144

8.2.5 函数 RTC_SetDate	144	10.1.1 I2C 主要特点	173
8.2.6 函数 RTC_SetAlarm	145	10.1.2 I2C 功能描述	173
8.2.7 函数 RTC_ITConfig	146	10.2 I2C 库函数	183
8.3 实时时钟实例——万年历	147	10.2.1 函数 I2C_DeInit	183
8.3.1 实例目的	147	10.2.2 函数 I2C_Init	184
8.3.2 实例要求	147	10.2.3 函数 I2C_Cmd	185
8.3.3 硬件基础	147	10.2.4 函数 I2C_GenerateSTART	186
8.3.4 软件结构	147	10.2.5 函数 I2C_GenerateSTOP	186
8.3.5 实例代码	149	10.2.6 函数 I2C_AcknowledgeConfig	186
8.3.6 编译下载和调试	153	10.2.7 函数 I2C_OwnAddress2Config	187
第 9 章 通用 SPI 的一般应用	154	10.2.8 函数 I2C_DualAddressCmd	187
9.1 SPI 简介	154	10.2.9 函数 I2C_GeneralCallCmd	188
9.1.1 SPI 特征	154	10.2.10 函数 I2C_ITConfig	188
9.1.2 SPI 引脚描述	155	10.2.11 函数 I2C_SendData	189
9.1.3 数据传输模式	156	10.2.12 函数 I2C_ReceiveData	189
9.1.4 SPI 从模式	157	10.2.13 函数 I2C_Send7bitAddress	190
9.1.5 SPI 主模式	158	10.2.14 函数 I2C_ReadRegister	191
9.1.6 状态标志	158	10.2.15 函数 I2C_SoftwareResetCmd	191
9.1.7 利用 DMA 的 SPI 通信	159	10.2.16 函数 I2C_TransferHandling	192
9.1.8 SPI 中断	159	10.2.17 函数 I2C_GetFlagStatus	193
9.2 SPI 库函数介绍	159	10.3 I2C 读写 24C02 实例	194
9.2.1 函数 SPI_Init	160	10.3.1 实例要求	194
9.2.2 函数 SPI_Cmd	162	10.3.2 硬件基础	194
9.2.3 函数 SPI_I2S_ITConfig	163	10.3.3 24C02 器件介绍	194
9.2.4 函数 SPI_I2S_DMACmd	163	10.3.4 软件结构	196
9.2.5 函数 SPI_SendData8	164	10.3.5 实例代码	197
9.2.6 函数 SPI_ReceiveData8	164	10.3.6 编译下载和调试	200
9.2.7 函数 SPI_I2S_GetFlagStatus	165	第 11 章 ADC 的一般应用	201
9.2.8 函数 SPI_I2S_ClearFlag	165	11.1 ADC 功能介绍	201
9.2.9 函数 SPI_I2S_GetITStatus	166	11.1.1 ADC 主要特性	201
9.3 SPI 通信实例——发送数据	166	11.1.2 ADC 引脚和内部信号	202
9.3.1 实例要求	166	11.1.3 ADC 功能描述	202
9.3.2 硬件基础	166	11.1.4 转换的外部触发和触发极性 (EXTSEL、EXTEN)	206
9.3.3 软件结构	167	11.1.5 数据对齐	208
9.3.4 实例代码	168	11.1.6 低功耗特性	210
9.3.5 编译下载和调试	172	11.1.7 模拟窗口看门狗 (AWDEN、 AWDSGL、AWDCH、AWD_HTR/ LTR、AWD)	211
第 10 章 I2C 接口的一般应用	173		
10.1 I2C 简介	173		

11.1.8 温度传感器	212
11.1.9 电池电压监测	213
11.1.10 ADC 中断	213
11.2 实现 ADC 最佳精度	213
11.2.1 ADC 模块自身相关的误差	213
11.2.2 与环境相关的 ADC 误差	216
11.2.3 如何减小与外部环境相关的 ADC 误差	219
11.3 ADC 库函数	226
11.3.1 函数 ADC_DeInit	226
11.3.2 函数 ADC_Init	226
11.3.3 函数 ADC_Cmd	228
11.3.4 函数 ADC_DMACmd	228
11.3.5 函数 ADC_ITConfig	229
11.3.6 函数 ADC_GetCalibrationFactor ..	230
11.3.7 函数 ADC_StartOfConversion	230
11.3.8 函数 ADC_ChannelConfig	230
11.3.9 函数 ADC_StructInit	232
11.3.10 函数 ADC_Analog WatchdogCmd	232
11.3.11 函数 ADC_AnalogWatchdog ThresholdsConfig	233
11.3.12 函数 ADC_AnalogWatchdogSingle ChannelCmd	233
11.3.13 函数 ADC_AnalogWatchdogSingle ChannelConfig	234
11.3.14 函数 ADC_TempSensorCmd	235
11.3.15 函数 ADC_VrefintCmd	235
11.3.16 函数 ADC_VbatCmd	236
11.3.17 函数 ADC_GetConversionValue ..	236
11.3.18 函数 ADC_DMACmd	237
11.3.19 函数 ADC_DMAResquest ModeConfig	237
11.3.20 函数 ADC_GetFlagStatus	238
11.3.21 函数 ADC_GetITStatus	238
11.3.22 函数 ADC_ClearITPendingBit ..	239
11.4 ADC 数据采集实例 1——单通道 数据采集	239

11.4.1 实例要求	239
11.4.2 硬件基础	239
11.4.3 软件结构	240
11.4.4 实例代码	243
11.4.5 编译下载和调试	246
11.5 ADC 数据采集实例 2——周围 温度采集	246
11.5.1 实例要求	246
11.5.2 硬件基础	247
11.5.3 软件结构	247
11.5.4 实例代码	247
11.5.5 编译下载和调试	250

第 12 章 嵌入式闪存的基本操作 251

12.1 嵌入式闪存介绍 251

12.1.1 闪存主要特性	251
12.1.2 闪存模块组织	251
12.1.3 读保护	252
12.1.4 FLASH 写和擦除操作	253
12.1.5 存储保护	255
12.1.6 写保护	256

12.2 FLASH 固件库函数 257

12.2.1 函数 FLASH_SetLatency	257
12.2.2 函数 FLASH_PrefetchBufferCmd ..	258
12.2.3 函数 FLASH_Unlock	258
12.2.4 函数 FLASH_Lock	259
12.2.5 函数 FLASH_ErasePage	259
12.2.6 函数 FLASH_EraseAllPages	259
12.2.7 函数 FLASH_ProgramWord	260
12.2.8 函数 FLASH_ProgramHalfWord ...	260

12.3 FLASH 读写实例 261

12.3.1 实例要求	261
12.3.2 硬件基础	261
12.3.3 软件结构	261
12.3.4 实例代码	262
12.3.5 编译下载和调试	264

第 13 章 定时器的一般应用 265

13.1 定时器功能介绍 265

13.1.1 定时器主要功能	265	13.3.2 硬件基础	303
13.1.2 定时器功能描述	266	13.3.3 软件结构	303
13.1.3 调试模式	281	13.3.4 实例代码	306
13.2 定时器固件库	282	13.3.5 编译下载和调试	309
13.2.1 函数 TIM_DeInit	282	13.4 TIME 应用实例 2——使用一个定时器产生 4 路不同占空比 PWM ...	309
13.2.2 函数 TIM_TimeBaseInit	282	13.4.1 实例目的	309
13.2.3 函数 TIM_OC1Init	284	13.4.2 实例要求	310
13.2.4 函数 TIM_OC2Init	286	13.4.3 硬件基础	310
13.2.5 函数 TIM_OC3Init	286	13.4.4 软件结构	310
13.2.6 函数 TIM1_OC4Init	287	13.4.5 实例代码	312
13.2.7 函数 TIM_ICInit function	288	13.4.6 编译下载和调试	315
13.2.8 函数 TIM_BDTRConfig	289	13.5 TIME 应用实例 3——使用定时器产生 4 路不同占空比和频率的 PWM	315
13.2.9 函数 TIM_Cmd	291	13.5.1 实例目的	315
13.2.10 函数 TIM_CtrlPWMOutputs	292	13.5.2 实例要求	315
13.2.11 函数 TIM_ITConfig	292	13.5.3 硬件基础	315
13.2.12 函数 TIM_SelectInputTrigger	293	13.5.4 软件结构	315
13.2.13 函数 TIM_EncoderInterfaceConfig	293	13.5.5 实例代码	319
13.2.14 函数 TIM_ARRPreloadConfig	294	13.5.6 编译下载和调试	324
13.2.15 函数 TIM_CCPreloadControl	295		
13.2.16 函数 TIM_OC1PreloadConfig	295		
13.2.17 函数 TIM_OC2PreloadConfig	296		
13.2.18 函数 TIM_OC3PreloadConfig	296		
13.2.19 函数 TIM_OC4PreloadConfig	297		
13.2.20 函数 TIM_SelectOutputTrigger	297		
13.2.21 函数 TIM_SelectSlaveMode	298		
13.2.22 函数 TIM_SelectMasterSlaveMode	298		
13.2.23 函数 TIM_SetCounter	299		
13.2.24 函数 TIM_SetAutoreload	299		
13.2.25 函数 TIM_GetCounter	300		
13.2.26 函数 TIM_GetPrescaler	300		
13.2.27 函数 TIM_GetFlagStatus	301		
13.2.28 函数 TIM_ClearFlag	302		
13.2.29 函数 TIM_GetITStatus	302		
13.2.30 函数 TIM_ClearITPendingBit	302		
13.3 TIME 应用实例 1——简单定时器应用	303		
13.3.1 实例要求	303		
		第 14 章 独立看门狗和窗口看门狗定时器	325
		14.1 独立看门狗一般特性介绍	325
		14.1.1 独立看门狗 (IWDG) 主要功能	325
		14.1.2 独立看门狗 (IWDG) 功能描述	325
		14.2 窗口看门狗一般特性介绍	326
		14.2.1 窗口看门狗 (WWDG) 主要特性	327
		14.2.2 窗口看门狗 (WWDG) 功能描述	327
		14.2.3 如何设置看门狗超时	328
		14.2.4 调试模式	329
		14.3 独立看门狗库函数介绍	329
		14.3.1 函数 IWDG_WriteAccessCmd	329
		14.3.2 函数 IWDG_SetPrescaler	329
		14.3.3 函数 IWDG_SetReload	330
		14.3.4 函数 IWDG_ReloadCounter	331

14.3.5 函数 IWDG_Enable.....	331	15.2.4 函数 DAC_SetChannel1Data	348
14.4 窗口看门狗库函数介绍	331	15.2.5 函数 DAC_ITConfig	348
14.4.1 函数 WWDG_DeInit.....	331	15.3 DAC 输出电压实例.....	349
14.4.2 函数 WWDG_SetPrescaler.....	332	15.3.1 实例要求	349
14.4.3 函数 WWDG_SetWindowValue... 332		15.3.2 硬件基础	349
14.4.4 函数 WWDG_EnableIT	333	15.3.3 软件结构	349
14.4.5 函数 WWDG_SetCounter	333	15.3.4 实例代码	350
14.4.6 函数 WWDG_Enable	334	15.3.5 编译下载和调试.....	353
14.4.7 函数 WWDG_GetFlagStatus.....	334	第 16 章 DMA 的一般应用.....	354
14.4.8 函数 WWDG_ClearFlag	335	16.1 DMA 简介.....	354
14.5 独立看门狗实例.....	335	16.1.1 DMA 主要特性	354
14.5.1 实例目的	335	16.1.2 DMA 功能描述	354
14.5.2 实例要求	335	16.2 DMA 库函数.....	358
14.5.3 硬件基础	336	16.2.1 函数 DMA_DeInit.....	358
14.5.4 软件结构	336	16.2.2 函数 DMA_Init.....	358
14.5.5 实例代码	336	16.2.3 函数 DMA_Cmd.....	361
14.5.6 编译下载和调试	338	16.2.4 函数 DMA_GetCurrDataCounter ...	362
14.6 窗口看门狗实例.....	338	16.2.5 函数 DMA_ITConfig.....	362
14.6.1 实例目的	338	16.2.6 函数 DMA_GetFlagStatus.....	363
14.6.2 实例要求	338	16.2.7 函数 DMA_ClearFlag.....	364
14.6.3 硬件基础	338	16.2.8 函数 DMA_GetITStatus	364
14.6.4 软件结构	339	16.2.9 函数 DMA_ClearITPendingBit	365
14.6.5 实例代码	339	16.3 DMA 应用实例 1——串口发送	
14.6.6 编译下载和调试	341	数据.....	365
14.7 本章小结.....	342	16.3.1 实例要求	365
第 15 章 DAC 的应用.....	343	16.3.2 硬件基础	365
15.1 DAC 简介	343	16.3.3 软件结构	366
15.1.1 DAC 主要特性	343	16.3.4 实例代码	367
15.1.2 使能 DAC 输出缓存	344	16.3.5 编译下载和调试.....	369
15.1.3 DAC 数据格式	344	16.4 DMA 应用实例 2——DMA 中断	
15.1.4 DAC 转换	344	接收数据.....	369
15.1.5 DAC 输出电压	345	16.4.1 实例要求	369
15.1.6 选择 DAC 触发	345	16.4.2 硬件基础	369
15.1.7 DMA 请求	345	16.4.3 软件结构	369
15.2 DAC 外设库函数	346	16.4.4 实例代码	369
15.2.1 函数 DAC_DeInit.....	346	16.4.5 编译下载和调试.....	371
15.2.2 函数 DAC_Init	346	16.5 DMA 应用实例 3——使用 SPI 产	
15.2.3 函数 DAC_Cmd	347	生 FM0 编码.....	372

16.5.1	实例要求	372
16.5.2	FM0 编码基础	372
16.5.3	编码算法	372
16.5.4	实例代码	373
16.5.5	编译下载和调试	377
16.6	DMA 应用实例 4——使用 DMA 采集多路电压	377
16.6.1	实例要求	377
16.6.2	硬件基础	378
16.6.3	软件结构	378
16.6.4	实例代码	378
16.6.5	编译下载和调试	382

第 17 章 串行 FLASH 数据储存方案

17.1	串行 FLASH 概述	383
17.1.1	SST25VF016B 概述	383
17.1.2	SST25VF016B 引脚说明	384
17.1.3	SST25VF016B 接口电路	384
17.2	API 软件包	385
17.2.1	软件包结构	385
17.2.2	SPI 初始化	385
17.2.3	读数据 API	388
17.2.4	写数据 API	389

17.2.5	FLASH 擦除 API	391
17.2.6	读 ID	393
17.3	FLASH 数据读写实例	393
17.3.1	实例目的	394
17.3.2	实例要求	394
17.3.3	硬件基础	394
17.3.4	实例代码	394
17.3.5	编译下载和调试	396

第 18 章 LCD 模块应用

18.1	LCD 模块介绍	397
18.1.1	LCD 屏性能特性	397
18.1.2	ILI9163C 驱动芯 SPI 时序	398
18.1.3	ILI9163C 寄存器介绍	399
18.2	LCD 模块相关 API 函数集	399
18.2.1	SPI 接口初始化	400
18.2.2	初始化 LCD 屏	402
18.2.3	LCD 屏 API 软件函数集	402
18.3	LCD 屏显示实例	410
18.3.1	实例目的	410
18.3.2	实例要求	410
18.3.3	硬件基础	410
18.3.4	软件结构	410
18.3.5	实例代码	410

第 1 章 STM32F051xx 系列芯片简介

1.1 STM32F051xx 系列芯片功能简介

STM32F051xx 系列芯片采用高性能的 ARM CortexTM-M0 的 32 位 RISC 内核（最高运行速度达 48MHz）、高速的嵌入式闪存（FLASH 最大 64KB，SRAM 最大 8KB），并广泛集成增强型外设和 I/O 口。所有器件提供标准的通信接口（最多两个 I2C、两个 SPI、一个 I2S、一个 HDMI CEC、两个 USART）、一个 12 位 ADC、一个 12 位 DAC、最多 5 个通用 16 位定时器、一个 32 位定时器和一个高级控制 PWM 定时器。

STM32F051xx 系列工作在 $-40^{\circ}\text{C} \sim +85^{\circ}\text{C}$ 或 $-40^{\circ}\text{C} \sim +105^{\circ}\text{C}$ 温度范围及 2.0V~3.6V 电源电压。一套全面的为低功耗应用设计准备的省电模式。

STM32F051xx 系列包括 3 种不同的封装，从 32 引脚到 64 引脚不等。根据选择的器件，包含不同数量的外设。

这些特点使得 STM32F051xx 微控制器系列应用广泛，如应用在控制和用户界面、手持设备、A/V 接收机和数字电视、PC 外设、游戏和 GPS 平台、工业应用、可编程控制器、逆变器、打印机、扫描仪、报警系统、视频对讲、HVACs 等。

STM32F051xx 的具体特点如下：

- ✧ 电压范围为 2.0V~3.6V。
- ✧ 内核为 ARM 32-bit Cortex[®]-M0 CPU（48 MHz max）。
- ✧ 存储大小为 32~64KB FLASH memory 和 8KB SRAM 带硬件校验。
- ✧ 具有 CRC 计算单元。
- ✧ 时钟管理有 4~32MHz 外部高速晶体振荡器、32.768kHz 可校准外部低速振荡器、内部 8MHz RC 带 x6 锁相环倍频、内部 32kHz RC 振荡器。
- ✧ 日历型 RTC 集成闹钟可周期性自动从 Stop/Standby 状态唤醒。
- ✧ 复位和供电管理有上电/掉电复位（POR/PDR）、可编程电压检测器（PVD）。
- ✧ 具有低功耗休眠、停止和待机模式。
- ✧ RTC 和备份区域 VBAT 单独供电。
- ✧ 具有 5 通道 DM 控制器。
- ✧ 具有 1×12 位、1.0 微秒 ADC（多至 16 采样通道），转换范围为 0V~3.6V，单独的 2.4V~3.6V 模拟供电。
- ✧ 具有两个高速低功耗模拟比较器，可编程输入/输出。
- ✧ 具有一个 12 位、D/A 转换器。
- ✧ 具有 55 个高速 I/O 口，全部可映射为外部中断输入；36 个 I/O 口支持 5V 容忍。
- ✧ 具有 11 个定时器。一个 16 位 7 通道高级控制定时器，用于 6 通道 PWM 输出，带死区时间发生器和紧急刹车功能；一个 32 位和一个 16 位定时器，每个有 4 路输入

捕获或输出比较通道，可用于红外控制和解码；一个 16 位定时器，带 2 通道输入捕获和输出比较及 1 对反极性输出通道，支持死区时间发生器和紧急刹车功能；两个 16 位定时器，都带输入捕获和输出比较及 1 对反极性输出通道，支持死区时间发生器和紧急刹车功能，支持 IR 控制调制门；一个 16 位定时器，带一路输入捕获/输出。

- ✧ 独立的窗口看门狗定时器。
- ✧ SysTick 定时器：24 位向下计数。
- ✧ 一个 16 位基本定时器，用于驱动 DAC。
- ✧ 通信接口，两个 I2C 接口，其中一个支持快速脉冲模式（1Mbit/s），20mA 灌电流，SMBus/PMBus 和从 STOP 状态唤醒；两个同步/异步串口，支持主同步 SPI 和 modem 控制功能，其中一个支持 ISO7816 接口、LIN、IrDA、自动波特率检测和唤醒功能；两个 SPI（18Mbit/s）外设，支持 4~16 位可编程字长，其中一个支持 I2S 接口复用。
- ✧ 具有消费电子控制（HDMI CEC）接口，具有帧头接收唤醒功能。
- ✧ 具有 18 个电容感应通道，支持接近、触摸按键、线性和旋转触摸传感器系列型号。
- ✧ 具有 96 位唯一 ID。
- ✧ 具有串行两线调试（SWD）。

STM32F051xx 系列器件的功能和外设数量如表 1-1 所示。

表 1-1 STM32F051xx 器件功能和外设数量表

外 围 设 备		STM32F051Kx			STM32F051Cx			STM32F051Rx		
FLASH (Kbytes)		16	32	64	16	32	64	16	32	64
SRAM (Kbytes)		4		8	4		8	4		8
定 时 器	高级	TIME1 (16 位)								
	通用	TIM2 (32 位)、TIM3 (16 位)、TIM14 (16 位)、TIM15 (16 位)、TIM16 (16 位)、TIM17 (16 位)								
	基本	TIM6 (16 位)								
通 信 接 口	SPI (I2S)	1		2	1		2	1		2
	I2C	1		2	1		2	1		2
	USART	1	2		1	2		1	2	
	CEC	1								
12 位 ADC		10 外部通道+3 内部通道						16 外部通道+3 内部通道		
GPIOs		27			39			55		
电容传感器		14			17			18		
12 位 DAC		1								
模拟比较器		2								
CPU 频率		48MHz								
工作电压		2.0V~3.6V								
工作温度		工作环境温度：-40℃~85℃/-40℃~105℃ 结温：-40℃~125℃								
封装		UFQFPN32			LQFP48			LOFP64		

1.2 功能概述

1.2.1 内核

STM32F051xx 系列 ARM 的 CortexTM-M0 处理器是 ARM 处理器中针对嵌入式系统的最新一代产品。它提供了一种低成本的平台旨在满足少引脚数和低功耗单片机的需求，同时提供出色的计算性能和先进的系统响应中断。

ARM 的 CortexTM-M0 的 32 位 RISC 处理器，提供卓越的代码效率，提供 ARM 内核的高性能预期，区别于同等的内存大小的 8 位和 16 位器件。STM32F051xx 系列采用嵌入式的 ARM 内核，因此与所有的 ARM 工具和软件兼容。

1.2.2 存储器

存储器具有以下特点：

- ◇ 多达 8KB 的嵌入式 SRAM，可用 CPU 的速度进行无等待位的读写访问，并包含针对高可靠性应用需要的嵌入式校验检查功能。
- ◇ 非易失性内存被分为两个区域：16~64B 的程序和数据嵌入式闪存区域和选项字节区域。选项字节用于对内存（4KB 的粒度）进行写保护设置和（或）对整个内存进行读出保护设置。
 - 0 级：没有读出保护。
 - 1 级：FLASH 读保护，不允许在调试功能连接时或从 RAM 启动时对 FLASH 进行读写操作。
 - 2 级：芯片读保护，完全禁止调试功能（Cortex-M0 的串行线）和从 RAM 启动。

1.2.3 循环冗余校验计算单元（CRC）

CRC 计算单元可以用来按照既定的多项式算法，依据输入数据快速算出循环冗余校验的结果码。在很多应用中，通常使用循环冗余校验的技术来检查数据传输或存储的完整性。在 EN/IEC 60335-1 功能安全标准范围内，其提供了校验 FLASH 存储可靠性的技术手段。CRC 计算单元可随时计算软件签名，使得可以在通信和存储时就地完成签名比较。

1.2.4 直接存储器访问控制器（DMA）

5 通道通用 DMA 可以管理存储器到存储器、外设到存储器和存储器到外设的直接访问。DMA 支持环形缓冲区的管理，在控制器达到缓冲区的末尾时不再需要用户代码的干预。

每个通道连接到专用硬件 DMA 请求，支持软件对每个通道的触发。由软件完成 DMA 的配置，源和目标之间传输的数据量都是独立的。DMA 可以用于主要的外设：SPI、I2S、