



CD-ROM

配光盘

(源程序和课件)

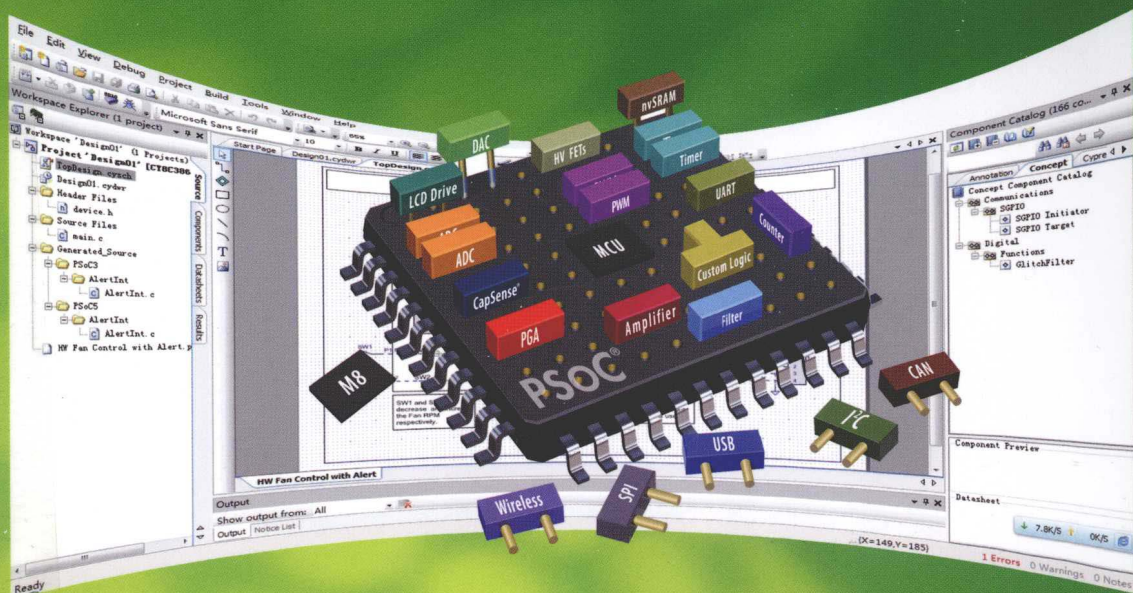
Cortex-M3

可编程片上系统

原理及应用

The Principle and Application of
Cortex-M3 Programmable System-on-Chip

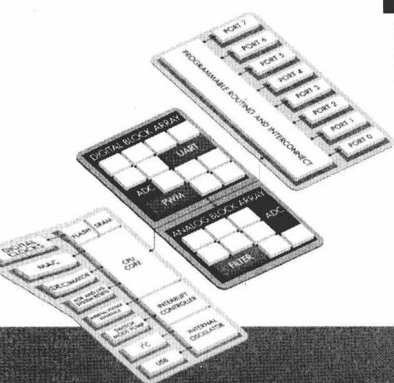
何 宾 编著



化学工业出版社

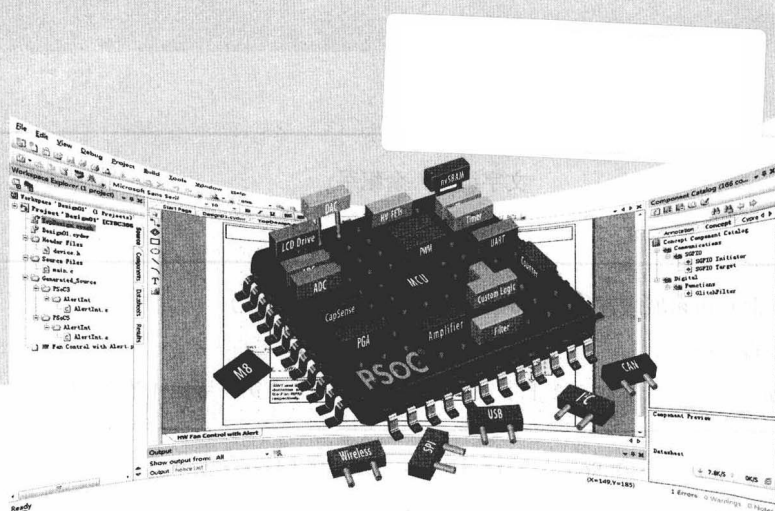


Cortex-M3 可编程片上系统 原理及应用



The Principle and Application of
Cortex-M3 Programmable System-on-Chip

何 宾 编著



CD-ROM
配光盘

(源程序和课件)



化学工业出版社

· 北 京 ·

图书在版编目 (CIP) 数据

Cortex-M3 可编程片上系统原理及应用 / 何宾编著. —北京: 化学工业出版社, 2012.7

ISBN 978-7-122-14451-5

I. ①C… II. ①何… III. ①微处理器-系统设计
IV. ①TP332

中国版本图书馆 CIP 数据核字 (2012) 第 117434 号

责任编辑: 宋 辉

文字编辑: 余纪军

责任校对: 宋 夏

装帧设计: 韩 飞

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

印 装: 大厂聚鑫印刷有限责任公司

787mm×1092mm 1/16 印张 20³/₄ 字数 532 千字 2012 年 10 月北京第 1 版第 1 次印刷

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

网 址: <http://www.cip.com.cn>

凡购买本书, 如有缺损质量问题, 本社销售中心负责调换。

定 价: 78.00 元 (1CD-ROM)

版权所有 违者必究

序

Cortex-M3 可编程片上系统原理及应用

ARM Cortex-M 处理器是一系列可向上兼容的高能效、易于使用的处理器,这些处理器旨在帮助开发人员满足嵌入式应用的需要,主要包括以更低的成本提供更多功能、不断增加连接、改善代码重用和提高能效。

ARM Cortex-M 处理器已成为全球微控制器标准,许可给 40 个以上的 ARM 合作伙伴。Cypress 公司将 ARM 的 Cortex-M3 处理器集成在其 PSoC5 器件内,为客户提供了基于 ARM 处理器的单片系统解决方案,这种解决方案提高了设计的可能性,缩短了系统设计周期,降低了设计成本,极大地满足了市场对产品竞争力的要求。通过 Cortex-M3 强大的处理能力,充分发挥 PSoC 器件内的数模混合阵列的性能。

何宾老师的《Cortex-M3 可编程片上系统原理及应用》通过对 PSoC5 器件相关内容的详细介绍,系统介绍了 Cortex-M3 CPU 处理器的结构、Cortex-M3 CPU 处理器的指令集、Cortex-M3 编程和调试接口、Cortex-M3 和外设的接口以及基于 Cortex-M3 的 μ C-OS III 操作系统等内容,并且通过相关的设计实例,介绍了基于 ARM Cortex-M3 CPU 的嵌入式系统的开发流程。

相信该书的出版,会对国内从事 Cortex-M 系统处理器应用的广大读者有所借鉴,并且通过 PSoC5 这一单片系统设计平台,熟练地掌握基于 Cortex-M3 的嵌入式系统的设计流程和实现方法。

时 昕

ARM 中国区大学计划经理 时昕

2012 年 7 月

前言

Cortex-M3 可编程片上系统原理及应用

随着半导体技术的发展和芯片集成度的提高,越来越多的厂商开始提供在单芯片上实现复杂系统的解决方案,即基于 PSoC 的解决方案。这种解决方案提高了设计的可靠性,缩短了系统设计周期,降低了设计成本,极大地满足了市场对产品竞争力的要求。

美国 Cypress 公司率先在业界实现了完全意义上的 PSoC 解决方案,即在单芯片上实现了 MCU、数字和模拟系统的高度集成。PSoC 技术的不断发展,将大大推动电子系统设计方法的创新,并且对未来嵌入式系统设计领域带来深远的影响。Cypress 的 PSoC5 集成了 ARM 公司的 Cortex-M3 CPU 核。这种集成高性能 Cortex-M3 CPU 的片上可编程系统,极大地改善了片上系统的整体性能,拓宽了片上可编程系统的应用领域,进一步满足了不同用户的需求。

本书全面系统介绍了 Cypress 公司的 PSoC5 可编程片上系统体系结构和设计流程,让读者能更好地掌握 PSoC 的体系结构和实现方法。本书各部分的主要内容如下。

PSoC 设计导论部分的内容包括微控制器基础、可编程片上系统 PSoC 概述、PSoC5 设计流程、PSoC5 的结构及功能、PSoC5 器件概述等。

PSoC5 CPU 及存储子系统部分的内容包括 Cortex-M3 内核概述、嵌套向量中断控制器、高速缓存控制器、PHUB 和 DMA 控制器和 PSoC5 存储器系统等。

PSoC5 CPU 指令系统部分的内容包括 Cortex-M3 指令寻址模式、Cortex-M3 CPU 指令集、Cortex-M3 汇编语言编程模型等。

PSoC5 公共资源部分的内容包括时钟管理、电源管理、看门狗定时器、复位、I/O 系统和布线资源等。

PSoC 编程和调试接口功能部分的内容包括测试控制器、Cortex-M3 调试和跟踪、非易失性存储器编程等。

基于 PSoC Creator 的程序设计部分的内容包括 PSoC Creator 软件功能、汇编语言 GPIO 控制程序的设计、C 语言 GPIO 控制程序的设计和中断服务程序的设计等。

定时器、计数器和 PWM 模块部分的内容包括定时器模块特性、定时器模块结构、PWM 控制 LED 显示的实现等。

LCD 显示驱动模块部分的内容包括 LCD 的工作原理、LCD 驱动特性、LCD 驱动系统、段式 LCD 显示的实现等。

I2C 总线模块部分的内容包括 I2C 总线模块概述、I2C 总线实现原理、I2C 总线操作模式、I2C 模块通信的实现等。

USB 总线模块部分的内容包括 USB 总线模块概述、USB 模块结构、USB 模块工作条件、逻辑传输模式、PS/2 和 CMOS I/O 模式、USB 模块寄存器列表和 USB 人体输入设备的实现等。

通用数字块 UDB 部分的内容包括通用数字块概述、UDB 模块结构、交通灯控制电路的设计与实现等。

模拟前端模块部分的内容包括模拟比较器、运算放大器模块、可编程 SC/CT 模块、精密参考源、基于混频器的精确整流的实现等。

ADC 和 DAC 模块部分的内容包括 $\Delta - \Sigma$ ADC 模块、SAR ADC 模块、DAC 模块、基于 SAR ADC 的数字电压表实现、基于 $\Delta - \Sigma$ 的数字电压表实现、IDAC 值显示的实现等。

电容感应模块部分的内容包括电容感应模块的结构、电容感应 $\Delta - \Sigma$ 算法、电容触摸感应实现等。

数字滤波器模块部分的内容包括数字滤波器模块概述、数字滤波器模块结构和基于 DFB 的数字滤波器实现等。

μ C-OS/III 操作系统部分的内容包括 μ C-OS/III 操作系统概述、 μ C-OS/III 操作系统内核及功能、PSoC5 硬件系统的构建、添加中断代码到 ISR 中、配置引脚、运行设计、启动 μ C/Probe 工具等。

本书不仅可以作为大学信息类专业讲授可编程片上系统、嵌入式系统设计等相关课程的教学用书，也可以作为从事相关领域科研工作者的参考用书。

本书由何宾编著，王纲领、常晓磊、彭渤、杨光伟为本书编写提供了帮助。Cypress 公司中国区大学计划经理魏荣博士为本书提供了数据手册、技术参考资料、PSoC5 硬件开发平台资源。在编写 μ C-OS/III 系统这一章时，北京麦克泰软件技术有限公司相关人员为本书提供了 μ C-OS/III 操作系统相关的外文参考书籍。在此，一并感谢。

由于编者水平有限，编写时间仓促，书中难免有疏漏之处，敬请读者批评指正。

编著者

目 录

Cortex-M3 可编程片上系统原理及应用

第 1 章 PSoC 设计导论	1
1.1 微控制器基础	1
1.1.1 微控制器的涵义	1
1.1.2 微控制器编程语言	2
1.2 可编程片上系统 PSoC 概述	3
1.2.1 PSoC 发展概述	3
1.2.2 PSoC 设计方法	5
1.3 PSoC5 设计流程	7
1.3.1 硬件设计流程	8
1.3.2 软件设计流程	9
1.4 PSoC5 的结构及功能	10
1.4.1 PSoC5 系统结构概述	10
1.4.2 数字子系统结构及功能	10
1.4.3 模拟子系统结构及功能	13
1.4.4 输入/输出引脚功能	14
1.5 PSoC5 器件概述	15
1.5.1 PSoC5 引脚分布	15
1.5.2 PSoC5 器件分类和资源	16
第 2 章 PSoC5 CPU 及存储子系统	19
2.1 Cortex-M3 内核结构概述	19
2.1.1 Cortex-M3 内核结构特性	19
2.1.2 流水线结构	20
2.1.3 寄存器	20
2.1.4 操作模式	23
2.1.5 SysTick 定时器	23
2.1.6 存储器空间映射	23
2.1.7 异常及处理	26
2.2 嵌套向量中断控制器	26
2.2.1 中断控制器的特性	26

2.2.2	中断使能	28
2.2.3	中断优先级	29
2.2.4	电平/脉冲中断	30
2.2.5	中断的执行	30
2.2.6	PSoC5 中断特性	31
2.2.7	中断控制器和功耗模式	34
2.3	高速缓存控制器	35
2.4	PHUB 和 DMA 控制器	36
2.4.1	PHUB	36
2.4.2	DMA 控制器	38
2.4.3	访问 DMAC	44
2.4.4	DMAC 传输模式	45
2.4.5	PHUB 和 DMAC 寄存器列表	46
2.5	PSoC5 存储器系统	47
2.5.1	SRAM 存储器结构及功能	47
2.5.2	非易失性锁存器结构及功能	48
2.5.3	Flash 程序存储器结构及功能	49
2.5.4	EEPROM 存储器结构及功能	50
第 3 章	PSoC5 CPU 指令系统	51
3.1	Cortex-M3 指令寻址模式	51
3.2	Cortex-M3 CPU 指令集	53
3.2.1	Cortex-M3 指令集概述	53
3.2.2	CMSIS 函数	54
3.2.3	存储器访问指令	54
3.2.4	通用数据处理指令	61
3.2.5	乘法和除法指令	69
3.2.6	饱和指令	71
3.2.7	比特位操作指令	72
3.2.8	分支和控制指令	74
3.2.9	杂项操作指令	77
3.3	Cortex-M3 汇编语言编程模型	80
第 4 章	PSoC5 公共资源	82
4.1	时钟管理	82
4.1.1	内部振荡器	83
4.1.2	外部振荡器	84
4.1.3	DSI 时钟	86

4.1.4	相位锁相环	86
4.1.5	USB 时钟	86
4.2	电源管理	87
4.2.1	电源模式	87
4.2.2	电源监控	89
4.3	看门狗定时器	89
4.4	复位	90
4.4.1	复位模块功能介绍	90
4.4.2	复位源	91
4.5	I/O 系统和布线资源	92
4.5.1	I/O 系统特性	92
4.5.2	I/O 驱动模式	94
4.5.3	DSI 控制数字 I/O	97
4.5.4	模拟 I/O 引脚	99
4.5.5	LCD 驱动引脚	100
4.5.6	电容感应触摸引脚	100
4.5.7	SIO 功能和特性	100
4.5.8	上电时 I/O 配置	101
4.5.9	过电压容限	101
4.5.10	端口中断控制器单元	102
第 5 章	PSoC 编程和调试接口功能	104
5.1	测试控制器	104
5.1.1	测试控制器结构	104
5.1.2	SWD 接口规范	104
5.1.3	PSoC5 SWD 的特性	106
5.2	Cortex-M3 调试和跟踪	107
5.2.1	内核调试	108
5.2.2	系统调试	109
5.3	非易失性存储器编程	110
第 6 章	基于 PSoC Creator 的程序设计	112
6.1	PSoC Creator 软件功能	112
6.2	汇编语言 GPIO 控制程序的设计	113
6.2.1	创建和配置工程	113
6.2.2	查看和设置公共资源	114
6.2.3	编写 GPIO 汇编语言控制程序	115
6.3	C 语言 GPIO 控制程序的设计	119
6.3.1	使用 C 语言指针对 GPIO 端口控制	120
6.3.2	调用 API 函数对 GPIO 端口控制	120

6.3.3	PSoC5 的 SRAM 内函数定位	126
6.4	中断服务程序的设计	128
6.4.1	创建和配置工程	128
6.4.2	添加 IP 核资源到设计	128
6.4.3	IP 核参数配置和连接	129
6.4.4	中断服务程序的设计	131
6.4.5	下载并调试工程	133
第 7 章	定时器、计数器和 PWM 模块	134
7.1	定时器模块特性	134
7.2	定时器模块结构	134
7.2.1	时钟选择	135
7.2.2	使能/禁止模块	136
7.2.3	输入信号特性	136
7.2.4	操作模式	137
7.2.5	中断使能	141
7.2.6	寄存器列表	141
7.3	PWM 控制 LED 显示的实现	142
7.3.1	创建和配置工程	142
7.3.2	编写软件程序	146
7.3.3	编程及调试	147
第 8 章	LCD 显示驱动模块	148
8.1	LCD 的工作原理	148
8.1.1	LCD 物理结构	148
8.1.2	LCD 液晶分类	149
8.2	LCD 驱动特性	154
8.3	LCD 驱动系统	154
8.4	LCD 功能描述	155
8.4.1	LCD DAC	155
8.4.2	LCD 配置选项	156
8.4.3	LCD 驱动模块	156
8.4.4	UDB	159
8.4.5	DMA	159
8.5	段式 LCD 显示的实现	160
8.5.1	段式 LCD 的功能	160
8.5.2	段式 LCD 的参数配置	162
8.5.3	编写软件程序	167
8.5.4	编程及调试	169

第 9 章	I ² C 总线模块	170
9.1	I ² C 总线模块概述	170
9.2	I ² C 总线实现原理	171
9.2.1	I ² C 总线模块结构	171
9.2.2	典型的 I ² C 数据传输	171
9.2.3	I ² C 总线寄存器及操作	172
9.3	I ² C 总线操作模式	173
9.3.1	从模式	173
9.3.2	主模式	174
9.3.3	多主模式	175
9.4	I ² C 模块通信的实现	176
9.4.1	系统实现原理	176
9.4.2	创建和配置工程	177
9.4.3	编写软件程序	181
9.4.4	编程及调试	183
第 10 章	USB 总线模块	184
10.1	USB 总线模块概述	184
10.2	USB 模块结构	184
10.2.1	串行接口引擎 SIE	185
10.2.2	仲裁器	186
10.3	USB 模块工作条件	187
10.3.1	工作频率	187
10.3.2	工作电压	188
10.3.3	收发器	188
10.3.4	端点	188
10.3.5	传输类型	188
10.3.6	中断	189
10.4	逻辑传输模式	189
10.4.1	非 DMA 访问	190
10.4.2	手工 DMA 访问	190
10.4.3	控制端点的逻辑传输	193
10.5	PS/2 和 CMOS I/O 模式	194
10.6	USB 模块寄存器列表	194
10.7	USB 人体学输入设备的实现	195
10.7.1	人体接口设备的原理	195
10.7.2	创建和配置工程	200
10.7.3	编写软件程序	204
10.7.4	编程及调试	205

第 11 章 通用数字块 UDB 207

11.1 通用数字块概述.....	207
11.2 UDB 模块结构.....	208
11.2.1 PLD 模块结构及宏单元.....	208
11.2.2 数据通道模块.....	209
11.2.3 状态和控制模块.....	212
11.3 交通灯控制电路的设计与实现.....	213
11.3.1 交通灯设计原理.....	213
11.3.2 交通灯控制电路的设计.....	213
11.3.3 引脚分配.....	218
11.3.4 设计下载与测试.....	218
11.4 静态时序分析.....	218

第 12 章 模拟前端模块 220

12.1 模拟比较器.....	220
12.1.1 输入和输出接口.....	220
12.1.2 LUT.....	220
12.2 运算放大器模块.....	221
12.3 可编程 SC/CT 模块.....	223
12.3.1 单纯的放大器.....	224
12.3.2 单位增益.....	225
12.3.3 可编程增益放大器.....	225
12.3.4 互阻放大器.....	227
12.3.5 连续时间混频器.....	228
12.3.6 采样混频器.....	228
12.3.7 Δ - Σ 调制器.....	230
12.3.8 跟踪和保持放大器.....	231
12.4 精密参考源.....	231
12.5 基于混频器的精确整流实现.....	233
12.5.1 整流器设计原理.....	233
12.5.2 创建和配置工程.....	233
12.5.3 编写软件程序.....	237
12.5.4 编程及调试.....	237

第 13 章 ADC 和 DAC 模块 238

13.1 Δ - Σ ADC 模块.....	238
13.1.1 Δ - Σ ADC 模块概述.....	238
13.1.2 Δ - Σ ADC 结构.....	239

13.1.3 Δ - Σ ADC 操作模式	241
13.2 SAR ADC 模块	242
13.2.1 SAR ADC 模块概述	242
13.2.2 SAR ADC 模块的工作原理	242
13.3 DAC 模块	243
13.4 基于 SAR ADC 的数字电压表实现	246
13.4.1 创建和配置工程	246
13.4.2 编写软件程序	249
13.4.3 编程及调试	250
13.5 基于 Δ - Σ ADC 的数字电压表实现	251
13.5.1 创建和配置工程	251
13.5.2 编写软件程序	253
13.5.3 编程及调试	254
13.6 IDAC 值显示的实现	254
13.6.1 创建和配置工程	255
13.6.2 编写软件程序	258
13.6.3 编程及调试	258
第 14 章 电容感应模块	259
14.1 电容感应模块的结构	259
14.2 电容感应 Δ - Σ 算法	262
14.3 电容感应触摸的实现	263
14.3.1 创建和配置工程	263
14.3.2 编写软件程序	266
14.3.3 编程及调试	267
第 15 章 数字滤波器模块	268
15.1 数字滤波器模块概述	268
15.2 数字滤波器模块结构	269
15.2.1 控制器	269
15.2.2 FSM RAM	270
15.2.3 数据通道	272
15.2.4 地址计算单元	273
15.2.5 总线接口和寄存器描述	274
15.3 基于 DFB 的数字滤波器实现	276
15.3.1 系统结构概述	276
15.3.2 元件参数配置	276
15.3.3 DMA 配置向导	280
15.3.4 编写软件程序	282
15.3.5 编程及调试	283

第 16 章 μ C-OS/III 操作系统284

16.1 μ C-OS/III 操作系统概述	284
16.1.1 操作系统的作用	284
16.1.2 μ C-OS/III 操作系统内核特征	285
16.1.3 μ C/OS-III 操作系统文件的结构	288
16.1.4 μ C/OS-III 操作系统应用程序结构分析	289
16.2 μ C-OS/III 操作系统内核及功能	296
16.2.1 单任务和多任务处理	296
16.2.2 临界区	300
16.2.3 任务管理	300
16.2.4 准备列表	301
16.2.5 调度	301
16.2.6 上下文切换	302
16.2.7 中断管理	302
16.2.8 等待列表	303
16.2.9 时间管理	303
16.2.10 定时器管理	303
16.2.11 资源管理	304
16.2.12 消息传递	305
16.2.13 多个对象等待处理	307
16.2.14 存储器管理	308
16.3 PSoC5 硬件系统的构建	309
16.4 添加中断代码到 ISR 中	310
16.4.1 添加中断代码到 ProbeUART_TxISR.c 中	310
16.4.2 添加中断代码到 ProbeUART_RxISR.c 中	311
16.5 配置引脚	311
16.6 运行设计	312
16.7 启动 μ C/Probe 工具	313

参考文献316

PSoC 设计导论

Cypress 公司的可编程片上系统(Programmable System-on-Chip, PSoC)将微控制器、可编程逻辑阵列、模拟可编程阵列等资源集成在单芯片上,为电子系统的设计带来了前所未有的机遇。

本章主要介绍:微控制器基础;可编程片上系统 PSoC 概述;PSoC5 设计流程;PSoC5 的结构及功能;PSoC5 器件概述。本章的内容是对 PSoC 所涉及知识的整体概述,通过本章内容的学习,帮助读者从“系统”角度把握 PSoC 技术的本质,方便对后续内容的学习和理解。

1.1 微控制器基础

微控制器是指带有外设的微处理器系统,比如台式电脑的 CPU,它是一个微处理器系统。微控制器将响应来自 I/O 引脚、定时器、通信等的输入,同时通过对信息进行操作控制来产生合适的输出信号。

I/O 引脚使得微控制器能读取来自其他设备的按钮和状态信息,同时 I/O 引脚也能够输出信号用来打开灯、运行电机和驱动显示设备。

定时器、通信模块和数/模转换模块能使微控制器执行特殊的任务,比如与 PC 机进行通信,读取温度信息等。

从微观上说,微控制器是一个集成了成千上万电子开关的设备。正如编程的人目的是为了将复杂的操作简化为逻辑和算术运算来完成那样,微控制器的设计人员必须决定使用什么电子设备来完成这些任务,比如,晶体管, FET 和二极管等。大多数的微控制器工作在二进制系统下,比如“1”或“0”,逻辑高或逻辑低,开或关。

Cypress 的微控制器系统称为可编程片上系统(Programmable System-on-Chip, PSoC),那是因为单芯片上包含了 CPU 内核、足够的模拟子系统和数字子系统资源。因此,在实现一个系统时,几乎不需要外部的电路。

1.1.1 微控制器的涵义

如图 1.1 所示,微处理器系统的 CPU 通常需要和其他部件相连接,这样才能使其发挥作用。微处理器系统通常会使用到的功能部件包括以下内容。

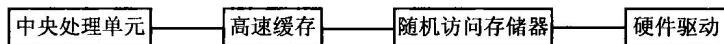


图 1.1 微处理器系统常用功能部件



(1) 中央处理单元 (CPU)

中央处理单元 (Central Processing Unit, CPU) 是系统的“大脑”，它知道如何和各种不同空间的存储器交换 (读或写) 信息。同时，也执行一些逻辑指令，最基本和最通用的有：加、减、逻辑“或”、逻辑“与”、逻辑“异或”、移位、移动和复制。一些处理器可能执行更加复杂的操作，但这些操作都是由最基本的操作组合得到的。

CPU 由一些子系统构成，在这些子系统中最重要的是程序计数器 (Program Counter, PC)，指令译码器和算术逻辑单元 (Arithmetic Logic Unit, ALU) 部分。PC 指向 Flash 存储器指定的地址，然后返回指令和数据。PC 用来确定送到指令译码器内的 Flash 中的字段。指令译码器包含译码逻辑，这些逻辑将对从 Flash 返回的数进行“翻译”，用来确定程序将执行的指令，这些指令将“告诉”CPU 下一步所做的逻辑操作行为。

CPU 不但能实现运算操作，也能修改程序运行的地址。如果在执行指令的过程中，并不是顺序的执行指令，比如遇到调转指令，那么 PC 将加载新的所要运行指令的地址，并且从指向 Flash 新的地址位置的地方执行程序。如果指令需要 CPU 执行一些运算，那么相关的数将送到 ALU 单元中。

此外，CPU 也能根据所接收到的指令对外设进行控制。

(2) 高速缓存 (Cache)

从位置和访问速度方面来说，高速缓存 Cache 最靠近 CPU。有时，将 Cache 直接集成在同一芯片内。但并不是必须放在同一个硅片上，只是封装在同一个芯片内。

(3) 随机访问存储器 (RAM)

从 CPU 访问速度来说，访问随机访问存储器 (Random Access Memory, RAM) 比访问高速缓存要慢。需要说明的是，这个词语已经失去了它的原本含义，这是由于现在大部分的存储器都能够以任何顺序进行访问。

(4) 硬件驱动

从速度来说，是系统中最慢和最大的存储部分。它用来保存程序，并且是由非易失性的存储介质构成。

1.1.2 微控制器编程语言

世界上不管是什么厂商的 CPU，不管采用什么样结构的 CPU，它们都有下面的共同特点：

- 都是靠程序计数器 (Program Counter, PC) 来控制程序的运行，正因为这个原因，它们本质上也是串行执行的；
- 工作在二进制状态下，也就是通常所说的在 PC 的控制下，通过运行二进制组成的机器代码，来控制 CPU 内各个功能部件的运行。

对 CPU 来说，所谓的“机器语言指令”，就是通过 CPU 内的控制逻辑来协调 CPU 内各个功能部件，完成所要求的操作。机器语言的运行效率是最高的。

机器语言指令应该由操作码和操作数两部分构成。操作码告诉 CPU 所需要执行的操作，操作数是执行操作所针对的对象。这些对象包括：立即数，寄存器和存储器等，通过访问这些对象来获得所需要操作的对象。比如：对 8051 来说，机器语言指令-7D25，表示该指令要实现数据传输操作，“7D”是操作码，“25”是操作对象，其表示将十六进制数#25，送到 R5 寄存器中。

但是，正如前面所说的，纯粹意义上的“机器语言”对程序员太难理解了，为什么？这是因为程序员是 CPU 的操作者，而不是 CPU 的设计者，程序员根本不可能从二进制代码的



排列中看出“机器语言”所描述的逻辑操作行为。

为了帮助程序员理解 CPU 所执行的操作,通过汇编语言助记符指令来帮助程序员设计指令以控制 CPU 的运行。汇编语言助记符指令通过汇编器被翻译成机器语言指令。用汇编助记符描述机器指令的形式为:

[标号:] 助记符 [操作数] [; 注释]

标号用来表示一行指令,助记符表示所要执行的逻辑操作行为,操作数为逻辑操作行为所操作具体对象,现在用汇编语言来描述上面的机器指令“7D25”:

MOV R5, #25

MOV 表示数据移动操作, R5 表示目的操作数, #25 表示源操作数,这个助记符汇编指令所表示的是,将立即数 25 复制到 R5 寄存器中。使用助记符来描述 CPU 所要进行的操作,比使用机器语言直接描述更加容易理解和记忆。但是,由于汇编语言下面是机器语言,所以对于使用汇编语言编程的程序员来说,他必须很清楚 CPU 的指令集,寄存器单元和存储器映射等繁琐的硬件规则。虽然其执行效率基本上和机器语言一样,但是使用汇编语言编程效率很低。因为很多程序员根本不了解 CPU 的具体内部结构,所以对他们来说,使用汇编语言编程并不比使用机器指令编程好到哪里去,这也是一件令他们非常痛苦的事情。

值得高兴的是,在今天,厂商开发的软件平台支持使用 C、BASIC 等高级语言对硬件进行编程; C 语言是不能直接在 CPU 上运行,它必须首先通过编译器(Compiler)转换成机器语言,才能在 CPU 上运行。使用高级语言所编写的代码其运行的效率不可能比用汇编语言编程的运行效率高,所以说,如果你想让 C 语言所编写的代码和汇编语言编写的代码有一样高的代码执行效率,你只能是绞尽脑汁的对 C 代码进行优化,或者使用 C 语言和汇编语言混合编程,来满足代码长度和运行时间的设计要求。虽然这一过程也会让程序员耗费很多的精力,但是,值得他们高兴的是,他们再也不用和底层硬件直接打交道了。

为了更清楚地说明 C 语言和汇编助记符之间的关系,列表 1.1 如下。

表 1.1 C 语言和汇编助记符之间的对应关系

C 语言描述	汇编助记符	机器指令	功 能
F=C+D	MOV A, 0x08	E508	将数据空间地址为 0x08 的内容送给累加器 A
	ADD A, 0x09	2509	将数据空间地址为 0x09 的内容和累加器 A 相加后送给累加器 A
	MOV 0x0A, A	F50A	将累加器 A 的内容送到地址为 0x0A 的数据空间

如今,越来越多的厂商提供了硬件的应用程序接口(Application Program Interface, API)函数,这样程序员可以根本不用知道更多的硬件实现细节,只需关心如何编写代码来使硬件工作,这样就大大提高了程序的设计效率。

1.2 可编程片上系统 PSoC 概述

1.2.1 PSoC 发展概述

当今世界,嵌入式处理器无处不在。在过去三十年间,由于一方面,市场要求不断地降低嵌入式系统的成本,另一方面要求嵌入式处理器处理及控制能力不断提高这样两个因素的推动,嵌入式处理器的功能变得越来越复杂。所以,当半导体市场出现越来越多的片上可编程系统,设计者就不会感到奇怪了。这种 PSoC 其实质就是将 CPU、模拟和数字子系统集成在单芯片上。最典型的,是世界著名的半导体厂商——美国 Cypress 公司,率先在单芯片上