



嵌入式软件工程方法与实践丛书

面向AMetal框架和 接口的C编程

周立功 主编

AMetal团队 参编



北京航空航天大学出版社
BEIHANG UNIVERSITY PRESS

方法与amp;实践丛书

面向 AMetal 框架和 接口的 C 编程

周立功 主编
AMetal 团队 参编

北京航空航天大学出版社

内 容 简 介

本书作为 AMetal 的基础教材,重点介绍 ZLG 在平台战略中所推出的 AMetal 开发平台。全书分为 4 个部分,第一部分由第 1 章组成,主要介绍 AM824_Core 开发套件,对微控制器和评估板进行了详细的介绍。第二部分由第 2~3 章组成,主要介绍模拟量与数字量的转换方法和相应的硬件电路设计。第三部分由第 4~8 章组成,重点介绍 AMetal 框架,包括接口的使用方法以及接口定义和实现的基本原理。第四部分由第 9~10 章组成,重点介绍基于 AMetal 无线硬件平台(包含 BLE 和 ZigBee)的通信和非常实用的 MVC 应用框架,并以开发温度检测仪为例,展示了程序设计和开发的详尽过程。

本书适合从事嵌入式软件开发、工业控制或工业通信的工程技术人员使用,也可作为大学本科、高职高专电子信息、自动化、机电一体化等专业学生的教学参考书,使学生在掌握 MCU 及各类外设使用方法的同时,还可以学习到在嵌入式开发中使用 C 实现面向对象的编程思想。

图书在版编目(CIP)数据

面向 AMetal 框架和接口的 C 编程 / 周立功主编. --
北京:北京航空航天大学出版社,2018.11
ISBN 978-7-5124-2872-0

I. ①面… II. ①周… III. ①C 语言—程序设计
IV. ①TP312.8

中国版本图书馆 CIP 数据核字(2018)第 261356 号

版权所有,侵权必究。

面向 AMetal 框架和接口的 C 编程

周立功 主编

AMetal 团队 参编

责任编辑 张冀青

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(邮编 100191) <http://www.buaapress.com.cn>

发行部电话:(010)82317024 传真:(010)82328026

读者信箱:emsbook@buaacm.com.cn 邮购电话:(010)82316936

涿州市新华印刷有限公司印装 各地书店经销

*

开本:710×1 000 1/16 印张:42 字数:895 千字

2018 年 11 月第 1 版 2018 年 11 月第 1 次印刷 印数:3 000 册

ISBN 978-7-5124-2872-0 定价:118.00 元

ZDS1104示波器

“小而美”的研发型示波器

小：专注于基础研发测试需求

美：追求极致的用户体验



28M
存储

17种
免费协议解码

100k
FFT分析

51种
参数测量

50k
波形刷新率

Touch
触屏操作



微信扫我 立即下单

前言

1. 存在的问题

最近十多年来,软件产业和互联网产业的迅猛发展,给人们提供了用武之地,同时也给软件工程教育带来了巨大的挑战。从教育现状看,通过灌输知识可以让人具有很强的考试能力,但却往往经不起用人单位的检验(笔试和机试)。虽然大家都知道,教育的本质在于培养人的创造力、好奇心、独特的思考能力和解决问题的能力,但实际上我们的教学实践有违教育理念。

代码的优劣不仅直接决定了软件的质量,还将直接影响软件的成本。软件的成本是由开发成本和维护成本组成的,而维护成本却远高于开发成本,蛮力开发的现象比比皆是,大量来之不易的资金被无声无息地吞没,整个社会资源浪费严重。为何不将复杂的技术高度抽象呢?如果能够实现,就能做到让专业的人做专业的事,AWorks 就是在这样的背景下诞生的。

2. 思维差别决定收益

其实,人与人之间的差别不在于知识和经验,而在于思维方式,思维方面的差异决定了每个人未来的不同。虽然大多数开发者都很勤奋,但其奋斗目标不是企业和个人收益最大化,而是以学习基础技术为乐趣,不愿意与市场人员和用户交流,不注重提升个人挖掘用户需求的能力,而是将精力用错了地方,这是很多人一辈子也没有认识到的深刻问题,而只是叹息自己怀才不遇,甚至将自己失败的责任推给他人。

一个软件系统封装了若干领域的知识,其中一个领域的知识代表了系统的核心竞争力,这个领域被称为“核心域”,其他领域称为“非核心域”。虽然更通俗的说法是“业务”和“技术”,但使用“核心域”和“非核心域”更严谨。

非核心域就是别人的领域,比如,底层驱动、操作系统和组件,即便你有一些优势,那也是暂时的,竞争对手也能通过其他渠道获得。非核心域的改进是必要的,但不充分,还是要在核心域上深入挖掘,让竞争对手无法轻易从第三方获得。因为在核



心域上深入挖掘,达到基于核心域的复用,这是获得和保持竞争力的根本手段。

要达到基于核心域的复用,有必要将核心域和非核心域分开考虑。因为过早地将各个领域的知识混杂,会增加不必要的负担,待解决问题的规模一旦变大,而人脑的容量和运算能力有限,从而导致开发人员腾不出脑力思考核心域中更深刻的问题,因此必须分而治之,因为核心域与非核心域的知识都是独立的。比如,一个计算器要做到没有漏洞,其中的问题也很复杂。如果不使用状态图对领域逻辑显式地建模,再根据模型映射到实现,而是直接下手编程,那么领域逻辑的知识靠临时去想,最终得到的代码肯定会破绽百出。其实有利润的系统,其内部都是很复杂的,千万不要幼稚地认为“我的系统不复杂”。

3. 利润到哪里去了?

早期创业时,只要抓住一个机会,多参加展会,多做广告,成功的概率就很大。在互联网时代,突然发现入口多了,聚焦用户的难度越来越大。当产品面临竞争时,你会发现“没有最低只有更低”;而且现在已经没有互联网公司了,“携程”变成了旅行社,“新浪”变成了新媒体……机会驱动、粗放经营的时代已经过去了。

Apple 公司之所以成为全球最赚钱的手机公司,关键在于产品的性能超越了用户的预期,且因为大量可重用的核心领域知识,将综合成本做到了极致。Yourdon 和 Constantine 在《结构化设计》一书中,将经济学作为软件设计的底层驱动力,软件设计应该致力于降低整体成本。人们发现软件的维护成本远远高于它的初始成本,因为理解现有代码需要花费时间,而且容易出错;同时,改动之后,还要进行测试和部署。

更多的时候,程序员不是在编码,而是在阅读程序。由于阅读程序需要从细节和概念上理解,因此修改程序的投入会远远大于最初编程的投入。基于这样的共识,让我们操心的一系列事情,需要不断地思考和总结使之可以重用,这就是方法论的源起。

通过财务数据分析可知,由于决策失误,我们开发了一些周期长、技术难度大且回报率极低的产品。由于缺乏科学的软件工程方法,不仅软件难以重用,而且扩展和维护难度很大,从而导致开发成本居高不下。

显而易见,从软件开发来看,软件工程与计算机科学是完全不同的两个领域的知识。其主要区别在于人,因为软件开发是以人为中心的过程。如果考虑人的因素,软件工程更接近经济学,而非计算机科学。如果不改变思维方式,则很难开发出既好卖成本又低的产品。

4. 解决之道

ZLG 集团(ZLG 集团目前包含两个子公司:广州致远电子有限公司, www.zlg.cn, 简称致远电子;广州周立功单片机科技有限公司, www.zlgmcu.com, 简称周立功单

片机。为便于描述,后文统一将 ZLG 集团简称为 ZLG)经过多年的理论探索和实践积累,开发了 AWorks,其中融入了更多的软件工程技术方法,目的是希望插上理想的翅膀,将程序员彻底从非核心域中释放出来聚焦于核心竞争力。无论你选择什么芯片和任何 OS,比如, Linux 和其他任何 RTOS,只要 AWorks 支持它,就可以在目标板上实现跨平台运行。因为无论什么 OS,它只是 AWorks 的一个组件,针对不同的 OS, AWorks 都会提供相应的适配器,那么所有的组件都可以根据需要更换。

由于 AWorks 制定了统一的接口规范,并对各种微处理器内置的功能部件与外围器件进行了高度的抽象;因此无论你选用的是 ARM 还是 DSP,通过“按需定制”的外设驱动软件和相关组件,以高度复用的软件设计原则和只对接口编程的思想为前提,则应用软件均可实现“一次编程,终生使用和跨平台”。基于此,进一步扩大了 AWorks 的使用范围,又发展出了代码量更少的 AMetal, AWorks 能给你带来的最大价值就是不需要重新发明轮子。

5. 丛书简介

这套丛书命名为《嵌入式软件工程方法与实践丛书》,目前已经完成《程序设计与数据结构》、《面向 AMetal 框架和接口的 C 编程》和《面向 AWorks 框架和接口的 C 编程(上)》,后续还将推出《面向 AWorks 框架和接口的 C 编程(下)》、《面向 AMetal 框架和接口的 LoRa 编程》、《面向 AWorks 框架和接口的 C++ 编程》、《面向 AWorks 框架和接口的 GUI 编程》、《面向 AWorks 框架和接口的 CAN 编程》、《面向 AWorks 框架和接口的网络编程》、《面向 AWorks 框架和接口的 EtherCAT 编程》和《嵌入式系统应用设计》等系列图书,最新动态详见 www.zlg.cn(致远电子官网)和 www.zlgmcu.com(周立功单片机官网)。

周立功

2018 年 8 月 13 日



第 1 章 AM824_Core 开发套件	1
1.1 LPC824 微控制器	1
1.1.1 特 性	1
1.1.2 概 述	2
1.2 LPC84x 微控制器	7
1.2.1 特 性	7
1.2.2 概 述	9
1.3 开关矩阵(SWM)	13
1.3.1 SWM 简介	13
1.3.2 SWM 应用	14
1.4 AM824_Core	16
1.4.1 电源电路	17
1.4.2 最小系统	18
1.4.3 复位与调试电路	19
1.4.4 板载外设电路	20
1.4.5 跳线帽的使用	23
1.4.6 MiniPort 接口	23
1.4.7 2×10 扩展接口说明	25
1.4.8 MicroPort 接口	26
1.5 MicroPort 模块介绍	28
1.5.1 SPI Flash 模块(MicroPort - Flash)	28
1.5.2 EEPROM 模块(MicroPort - EEPROM)	29
1.5.3 RTC 模块(MicroPort - RTC)	30
1.5.4 USB 模块(MicroPort - USB)	32
1.5.5 RX8025T 模块(MicroPort - RX8025T)	33
1.5.6 DS1302 模块(MicroPort - DS1302)	35
1.5.7 Analog 模块(MicroPort - Analog)	36



1.6	MiniPort 模块介绍	38
1.6.1	LED 模块(MiniPort - LED)	38
1.6.2	数码管模块(MiniPort - View)	40
1.6.3	按键模块(MiniPort - Key)	41
1.6.4	595 模块(MiniPort - 595)	43
1.6.5	ZLG72128 模块(MiniPort - ZLG72128)	44
第 2 章 ADC 信号调理电路设计		47
2.1	应用背景	47
2.1.1	标称精度	47
2.1.2	外围电路	48
2.1.3	干扰源	48
2.2	电路设计	49
2.2.1	基准源	50
2.2.2	低噪声模拟电源	52
2.2.3	瞬态驱动	53
2.2.4	输入信号滤波	55
2.2.5	模拟地与数字地	57
2.2.6	I/O 扇出电流	58
2.3	必要措施	58
2.3.1	输入范围匹配	59
2.3.2	多通道采样设置	60
2.3.3	电源分配策略	63
2.3.4	PCB 布局布线处理	63
2.4	实测验证	64
2.4.1	无噪声分辨率	64
2.4.2	积分非线性(INL)	65
2.4.3	失调与增益误差	68
2.5	应用说明	68
第 3 章 PWM 实现 DAC 电路设计		70
3.1	实现原理	70
3.1.1	PWM 信号时域分析	70
3.1.2	PWM 信号频域分析	71
3.2	电路设计	72
3.2.1	DAC 分辨率	72
3.2.2	有源低通滤波器	74
3.3	测试验证	75
3.3.1	DNL	75

3.3.2 INL	76
3.3.3 建立时间	77
3.4 参数总结	78
第4章 面向接口的编程	79
4.1 平台技术	79
4.1.1 创新的窘境	79
4.1.2 AWorks	80
4.1.3 AMetal	82
4.2 开关量信号	90
4.2.1 I/O 系统	90
4.2.2 输出控制	91
4.3 LED 数码管	97
4.3.1 静态显示	97
4.3.2 动态显示	100
4.3.3 闪烁处理	105
4.4 事件驱动	110
4.4.1 中断与事件驱动	110
4.4.2 软件定时器	111
4.5 键盘管理	120
4.5.1 独立按键	120
4.5.2 矩阵键盘	126
4.6 SPI 总线	133
4.6.1 SPI 总线简介	133
4.6.2 74HC595 接口	136
4.7 I ² C 总线	142
4.7.1 I ² C 总线简介	142
4.7.2 LM75B 接口	143
4.7.3 温控器	145
第5章 深入浅出 AMetal	151
5.1 接口与实现	151
5.1.1 GPIO 接口函数	151
5.1.2 LED 接口与实现	159
5.1.3 I/O 接口与中断	163
5.2 LED 数码管接口	167
5.2.1 静态显示	167
5.2.2 动态显示	172
5.2.3 代码重构	176



5.3 键盘扫描接口	186
5.3.1 单个独立按键	186
5.3.2 多个独立按键	190
5.3.3 矩阵键盘	195
5.4 PWM 接口	201
5.4.1 初始化	201
5.4.2 PWM 接口函数	202
5.4.3 蜂鸣器接口函数	204
5.5 SPI 总线	206
5.5.1 初始化	206
5.5.2 接口函数	207
5.5.3 SPI 扩展接口	216
5.6 I ² C 总线	225
5.6.1 初始化	225
5.6.2 接口函数	226
5.6.3 I ² C 扩展接口	229
5.7 A/D 转换器	232
5.7.1 模/数信号转换	232
5.7.2 初始化	236
5.7.3 接口函数	237
5.7.4 温度采集	240
5.8 UART 总线	253
5.8.1 初始化	254
5.8.2 接口函数	255
5.8.3 带缓冲区的 UART 接口	261
第 6 章 重用外设驱动代码	266
6.1 EEPROM 存储器	266
6.1.1 器件简介	266
6.1.2 初始化	267
6.1.3 读/写函数	270
6.1.4 NVRAM 通用接口函数	272
6.2 SPI NOR Flash 存储器	278
6.2.1 基本功能	278
6.2.2 初始化	279
6.2.3 接口函数	282
6.2.4 MTD 通用接口函数	285
6.2.5 FTL 通用接口函数	290
6.2.6 微型数据库	297

6.3	RTC 实时时钟	299
6.3.1	PCF85063	299
6.3.2	RTC 通用接口	302
6.3.3	闹钟通用接口	306
6.3.4	系统时间	312
6.3.5	特殊功能控制接口	317
6.3.6	RX8025T	318
6.3.7	DS1302	328
6.4	键盘与数码管接口	337
6.4.1	ZLG72128 简介	337
6.4.2	ZLG72128 初始化	346
6.4.3	按键管理接口函数	349
6.4.4	数码管显示接口函数	351
第 7 章	面向通用接口的编程	361
7.1	LED 控制接口	361
7.1.1	LED 通用接口	361
7.1.2	LED 驱动	363
7.1.3	MiniPort - LED	366
7.2	HC595 接口	368
7.2.1	HC595 通用接口	368
7.2.2	HC595 驱动	369
7.2.3	使用 HC595 驱动 LED	373
7.3	蜂鸣器控制接口	375
7.3.1	蜂鸣器通用接口	375
7.3.2	无源蜂鸣器驱动	377
7.4	温度采集接口	378
7.4.1	温度传感器通用接口	378
7.4.2	LM75B 驱动	379
7.5	键 盘	382
7.5.1	通用键盘接口	382
7.5.2	独立键盘驱动	385
7.5.3	矩阵键盘驱动	388
7.6	数码管	394
7.6.1	通用数码管接口	394
7.6.2	数码管驱动	398
7.6.3	数码管驱动(HC595 输出段码)	403
7.7	数码管与矩阵键盘联合使用	407
7.7.1	数码管、键盘与 I/O 驱动	407



7.7.2 数码管、键盘与 HC595 驱动	411
7.8 ZLG72128——数码管与键盘管理	415
7.8.1 ZLG72128 简介	415
7.8.2 ZLG72128 驱动	416
7.9 温控器	421
第 8 章 深入理解 AMetal	426
8.1 LED 通用接口	426
8.1.1 定义接口	426
8.1.2 实现接口	428
8.2 HC595 接口	444
8.2.1 定义接口	444
8.2.2 实现接口	446
8.3 蜂鸣器接口	454
8.3.1 定义接口	454
8.3.2 实现接口	455
8.4 温度采集接口	463
8.4.1 定义接口	463
8.4.2 实现接口	464
8.5 通用按键接口	469
8.5.1 定义接口	469
8.5.2 实现接口	472
8.5.3 检测按键的实现	480
8.6 通用数码管接口	486
8.6.1 定义接口	486
8.6.2 实现接口	488
第 9 章 BLE&ZigBee 无线模块	509
9.1 BLE 核心板	509
9.1.1 产品简介	509
9.1.2 协议说明	513
9.1.3 蓝牙模块初始化	514
9.1.4 蓝牙模块控制接口	518
9.1.5 蓝牙模块读/写数据接口	526
9.1.6 应用案例	529
9.2 ZigBee 核心板	531
9.2.1 产品简介	531
9.2.2 组网应用	534
9.2.3 ZigBee 初始化	535

9.2.4	ZigBee 配置接口	538
9.2.5	ZigBee 数据传输接口	545
9.2.6	应用案例	548
9.3	MVC 框架	551
9.3.1	MVC 模式	551
9.3.2	观察者模式	552
9.3.3	领域模型	555
9.3.4	子系统体系结构	556
9.3.5	软件体系结构	557
9.3.6	MVC 应用程序优化	579
第 10 章	温度检测仪	582
10.1	业务建模	582
10.1.1	问题描述	582
10.1.2	系统用例图	583
10.2	分析建模	584
10.2.1	领域词典	584
10.2.2	类模型	585
10.2.3	交互模型	586
10.2.4	按键处理模型	589
10.3	温度检测设计	590
10.3.1	子系统接口	590
10.3.2	设计模型	592
10.3.3	模型初始化	594
10.3.4	设置与获取数据	595
10.3.5	报警状态	598
10.4	视图设计	600
10.4.1	数码管视图	600
10.4.2	蜂鸣器视图	602
10.4.3	ZigBee 视图	604
10.5	按键处理模块设计	609
10.5.1	SET 键处理	609
10.5.2	INC(加)键处理	616
10.5.3	DEC(减)键处理	617
10.5.4	L/R 键处理	617
10.5.5	初始化	618
10.6	状态机设计	622
10.6.1	状态模型	622
10.6.2	设计模型	623



10.6.3 状态机	627
10.6.4 状态机接口	633
10.6.5 动作类	640
10.7 应用程序	651
参考文献	655

第 1 章

AM824_Core 开发套件

本章导读

随着物联网技术的发展,MCU 处理器的能力日益强大,如今的 MCU 与微处理器的界限越来越模糊,将会进一步融合成为嵌入式处理器。由于 AMetal 已经完全屏蔽了底层的复杂细节,因此开发者仅需了解 MCU 的基本功能就可以了。

1.1 LPC824 微控制器

1.1.1 特 性

- 系统:
 - ARM Cortex-M0+ 嵌入式处理器,内置可嵌套中断向量控制器(NVIC),系统节拍定时器,运行时频率高达 30 MHz;
 - 支持串行线调试(SWD)模式与 JTAG 边界扫描(BSDL)模式。
- 最高 32 KB 片内 Flash 和 8 KB SRAM,带 64 字节页面写入和擦除功能。
- 数字外设:
 - 集成了多达 32 个通用 I/O 引脚,并具备可配置上拉/下拉电阻、可编程开漏模式、输入反相器和干扰滤波器,GPIO 方向控制支持各个位的独立置位/清零/触发;
 - 4 个引脚具备 20 mA 的输出驱动能力,2 个开漏引脚具备 20 mA 灌入驱动能力;
 - GPIO 中断生成能力,8 个 GPIO 输入具有布尔模式匹配特性;
 - 开关矩阵,用于灵活配置每个 I/O 引脚功能;
 - CRC 引擎,带 18 个通道和 9 个触发输入的 DMA。
- 定时器:
 - 状态可配置定时器(SCTimer/PWM),输入和输出功能(包括捕获和匹配)用于定时和 PWM 应用;
 - 四通道多速率定时器(MRT),以多达 4 种可编程固定速率生成可重复中断;



- 自唤醒定时器(WKT),采用 IRC、低功耗、低频率内部振荡器作为时钟,或 always-on 电源域的外部时钟输入作为时钟;
- 窗口看门狗定时器(WWDI)。
- 模拟外设:
 - 一个 12 位 ADC,多达 12 个输入通道,带有多内部和外部触发输入,采样速率高达 1.2 MS/s,ADC 支持两个独立的转换顺序;
 - 比较器,带有 4 个输入引脚以及外部或内部基准电压。
- 串行接口:
 - 3 个 USART 接口,引脚功能通过开关矩阵和一个共用小数波特率发生器分配;
 - 2 个 SPI 控制器,引脚功能通过开关矩阵分配;
 - 4 个 I²C 总线接口,一个 I²C 支持高速模式 plus,在两个真开漏引脚和监听模式上,数据率为 1 Mbit/s,三个 I²C 支持标准数字引脚的数据率高达 400 kbit/s。
- 时钟生成:
 - 调整到 1.5%精度的 12 MHz 内部 RC 振荡器,可选择性地用作系统时钟;
 - 晶体振荡器,工作频率范围为 1~25 MHz;
 - 可编程看门狗振荡器,频率范围为 9.4~2.3 MHz;
 - 用于 WKT 的 10 kHz 低功耗振荡器;
 - PLL 使 CPU 无需使用高频晶体即可生成最高 CPU 主频,可从系统振荡器、外部时钟输入或内部 RC 振荡器运行;
 - 带分频器的时钟输出功能,可反映所有内部时钟源。
- 功率控制:
 - 可最大程度降低功耗的集成式 PMU(电源管理单元);
 - 节能模式:睡眠模式、深度睡眠模式、掉电模式和深度掉电模式;
 - 深度睡眠模式和掉电模式可由 USART、SPI 和 I²C 外设唤醒;
 - 深度掉电模式可由定时器控制进行自唤醒;
 - 上电复位(POR),掉电检测(BOD)。
- 单电源(1.8~3.6 V),工作温度范围为-40~+105 °C。

1.1.2 概述

LPC824 系列微控制器(MCU)具有丰富的片上外设,内部功能框图详见图 1.1。除 GPIO 外,还支持开关矩阵、状态可配置定时器、多速率定时器、窗口看门狗定时器和 DMA 控制器等。模拟外设包括 12 位高速 ADC 和模拟比较器,支持 3 路 UART、2 路 SPI 和 4 路 I²C。此外,芯片内部还集成了 12 MHz 的 RC 振荡器,可以作为系统的时钟源。