



嵌入式软件工程方法与实践丛书

面向AWorks框架和 接口的C编程（上）

周立功 主编

AWorks团队 参编



北京航空航天大学出版社
BEIHANG UNIVERSITY PRESS

嵌入式软件工程方法与实践丛书

面向 AWorks 框架和 接口的 C 编程(上)

周立功 主编

AWorks 团队 参编

北京航空航天大学出版社

内 容 简 介

本书作为 AWorks 的基础教材,重点介绍 ZLG 在平台战略中所推出的 AWorks 开发平台。全书分为 3 部分:第一部分为简介篇,由第 1 章组成,主要介绍 AWorks 的基本概念,包括 AWorks 诞生的背景、AWorks 架构及 AWorks 的重大意义等;第二部分为硬件篇,包括第 2~5 章,介绍了一系列支持 AWorks 的核心板及评估底板;第三部分为软件篇,包括第 6~15 章,重点介绍 AWorks 中基础服务的使用方法,主要包括常用设备(LED、按键、数码管等)、常用外设(GPIO、PWM、SPI 等)、时间管理、内存管理、OS 内核、文件系统,以及 AWBus-lite 框架,完整展示了 AWorks 中驱动的实现原理,并在最后介绍了一系列信号采集模块、接口扩展模块、常用外围器件等模块或器件的使用方法。

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、机电一体化等专业的教学参考书。

图书在版编目(CIP)数据

面向 AWorks 框架和接口的 C 编程. 上/周立功主编

. -- 北京:北京航空航天大学出版社,2018.11

ISBN 978-7-5124-2871-3

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 257888 号

版权所有,侵权必究。

面向 AWorks 框架和接口的 C 编程(上)

周立功 主编

AWorks 团队 参编

责任编辑 杨 昕

*

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:37.25 字数:794 千字

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

ISBN 978-7-5124-2871-3 定价:108.00 元

若本书有倒页、脱页、缺页等印装质量问题,请与本社发行部联系调换。联系电话:(010)82317024

A Works 的哲学思想

一、思维差异

苹果公司之所以能成为全球最赚钱的公司,关键在于其产品的性能超越了用户的预期,并且具备大量可重用的核心领域知识,使综合成本达到了极致。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,认为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,这是因为理解现有代码需要花费时间,且容易出错;而改动代码之后,还要进行测试和部署。

在很多时候,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,我们考虑一系列事情时,就要不断地总结使之可以重用,这就是方法论的源起。

通过财务数据分析可知,由于早期决策失误和缺乏科学的软件工程方法,我们开发了一些周期长、技术难度大且回报率极低的产品,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。

由此可见,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。显然,如果我们不改变思维方式,就很难开发出既好卖成本又低的产品。

二、利润模型

产品的 BOM 成本很低,而毛利又很高,为何很多上市公司的年利润却买不起一套房?房子到底被谁买走了,这个问题值得我们反思!

成功的企业除了愿景、使命和价值观之外,其核心指标就是利润。作为开发人员,最大的痛苦之一就是很难精准地开发出好卖的产品。因为很多企业都不知道利润是如何来的,所以有必要建立一个利润模型,即“利润=需求-设计”。需求是致力



于解决“产品如何好卖”的问题,设计是致力于解决“如何降低成本”的问题。

代码的优劣不仅直接决定了软件的质量,还将直接影响软件成本。软件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,造成社会资源的严重浪费。

为何不将复杂的技术高度抽象呢?如果实现了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。由于其中融入了更多的软件工程技术方法,因此就能做到将程序员彻底从非核心域中解脱出来,专注于核心竞争力。

三、核心域和非核心域

其实一个软件系统封装了若干领域的知识,其中有一个领域的知识代表了系统的核心竞争力,则这个领域就称为“核心域”,而其他领域就称为“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。虽然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。因为只有核心域上深入挖掘,达到基于核心域的复用,才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,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。由于核心域与非核心域的知识都是独立的,解决问题的规模一旦变大,而人脑的容量和运算能力又有限,就会顾此失彼,故必须分而治之。

四、共性与差异性

如果没有 ARM 公司的 IP 授权模式,那么在设计 MCU 时势必会消耗大量来之不易的财富。虽然 ARM 公司的规模相对来说不大,但是毫不影响 ARM 成为一个伟大的企业,其为人类做出的贡献是有目共睹的。

尽管如此,如果没有软件的支持,那么硬件就是一坨废铁。由于需求五花八门,人们尽管也做出了巨大的努力,期望最大限度地降低开发成本,但期望的实现有,却遥遥无期,无法做到高度地重用人类通过艰苦努力积累的知识。由于商业利益的驱使,伟大企业的不伟大之处,是企图将客户绑在他们的战车上,让竞争对手绝望,大凡成功的企业无不如此。

有没有破解的办法呢?有,那就是“共性与差异性分析”抽象工具。实际上,不管是基于何种内核的 MCU,也不管是哪家公司的 OS,其设计原理都是一样的,只是实现方法和实体(硬件和程序)不一样,但只要将其共性抽象为统一接口,差异性用特殊的接口应对即可。

基于此,我们不妨做一个大胆的假设。虽然 PCF85063、RX8025T 和 DS1302 来

自不同的半导体公司,但其共性都是 RTC 实时日历时钟芯片,即可高度抽象共用相同的驱动接口,其差异性可用特殊的驱动接口应对。虽然 FreeRTOS 或 $\mu\text{C}/\text{OS}-\text{II}$ 或 sysBIOS、Linux、Windows 各不相同,但它们都是 OS,多线程、信号量、消息、邮箱、队列等是其特有的共性,显然 QT 和 emWin 同样可以高度抽象为 GUI 框架。也就是说,不管是什么 MCU,也不管是否使用操作系统,只要修改相应的头文件,即可复用应用代码。

由此可见,无论选择何种 MCU 和 OS,只要 AWorks 支持它,就可以在目标板上实现跨平台运行。因为无论何种 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互换。

由于 AWorks 制定了统一的接口规范,并对各种 MCU 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,只要以高度复用的软件设计原则和只针对接口编程的思想为前提,应用软件就可实现“一次编程、终生使用、跨平台”,显然 AWorks 所带来的最大价值就是不需要重新发明轮子。

五、生态系统

如果仅有 OS 和应用软件框架,要构成生态系统,这是远远不够的。在万物互联的时代,一个完整的 IoT 系统,还包括传感器、信号调理电路、算法和接入云端的技术,可以说异常复杂,包罗万象。这不是一个公司拿到需求就可以在几个月之内完成的,需要长时间的大量积累。

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和广州周立功单片机科技有限公司,后文将 ZLG 集团简称为 ZLG,两个子公司分别简称为致远电子和周立功单片机)在成立之初就做了长远的布局,并没有将自己定位于芯片代理或设计,也没有将自己定位于仪器制造,更没有将自己定位于方案供应商,而是随着时间的推移和时代的发展,经过艰苦的努力自然而然地成为了“工业互联网生态系统”的领导品牌。这不是刻意为之的,而是通过长期的奋斗顺理成章的结果。

ZLG 通过“芯片+AWorks”设计了高附加值的模块、板卡和高端测量仪器,通过有线和无线通信接口接入 ZWS(ZLG Web Services) IoT 云端处理系统,实现了大数据处理,构成工业互联网生态系统。

ZLG 的商业模式既可以销售硬件,也可以销售平台,还可以针对某个特定的行业提供系统服务于终端用户。与此同时,ZLG 将在全国 50 所大学建立工业互联网生态系统联合实验室,通过产学研的模式培养人才服务于工业界,还将通过天使投资打造 ZLG 系,帮助更多的人取得更大的成功,推动“中国制造 2025”计划的高速发展。

六、专家与通才

任何一个组织和系统的成功都离不开专家和通才的鼎力配合与奋斗。这 12 年



一路走来很不容易,但欣慰的是 AWorks 生态系统的开发,培养了一些专家和核心骨干人才。我深深地体会到卓越人才的价值,多么高的评价都不为过,所以今后我的主要工作就是寻找和发现卓越人才,为他们提供一个衣食无忧的,能够静下心来创新的平台,大家共同努力改变这世界的某一小部分。欢迎自我推荐或读者向我推荐有理想有抱负的人才,如果你有心就一定能找到我。

七、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(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》《嵌入式系统应用设计》等图书,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 5 月 4 日



第一部分 简介篇

第1章 AWorks 简介	2
1.1 诞生的背景	2
1.2 基本概念	4
1.3 硬件平台	5
1.4 AWorks OS	7
1.4.1 特点	8
1.4.2 架构	9
1.4.3 发布形式	12
1.4.4 使用方法	13
1.5 云接入	14

第二部分 硬件篇

第2章 Cortex-M 系列无线核心板	16
2.1 M105x 无线核心板(M7 核)	16
2.1.1 概述	16
2.1.2 产品特性	17
2.1.3 产品功能框图	18
2.1.4 产品选型	18
2.1.5 I/O 信息	19
2.2 AW54101 无线核心板(M4 核)	24
2.2.1 概述	24
2.2.2 产品特性	24
2.2.3 产品功能框图	25



2.2.4	产品选型	25
2.2.5	I/O 信息	26
2.3	AW412 无线核心板(M4 核)	28
2.3.1	概述	28
2.3.2	产品特性	28
2.3.3	产品功能框图	29
2.3.4	产品选型	29
2.3.5	I/O 信息	30
2.4	Cortex-M 系列无线核心板快速选型	32
第 3 章 ARM9、DSP、Cortex-A 系列核心板		34
3.1	A280 核心板(ARM9 核)	34
3.1.1	概述	34
3.1.2	产品特性	35
3.1.3	产品功能框图	35
3.1.4	产品选型	36
3.1.5	I/O 信息	36
3.2	AW6748 核心板(DSP 核)	42
3.2.1	概述	42
3.2.2	产品特性	43
3.2.3	产品功能框图	44
3.2.4	产品选型	44
3.2.5	I/O 信息	45
3.3	M28x-T 核心板(ARM9 核)	51
3.3.1	概述	51
3.3.2	产品特性	52
3.3.3	产品功能框图	53
3.3.4	产品选型	53
3.3.5	I/O 信息	54
3.4	M6G2C 核心板(A7 核)	60
3.4.1	概述	60
3.4.2	产品特性	60
3.4.3	产品功能框图	61
3.4.4	产品选型	61
3.4.5	I/O 信息	62
3.5	M3352 核心板(A8 核)	67
3.5.1	概述	67
3.5.2	产品特性	68
3.5.3	产品功能框图	68

3.5.4 产品选型	69
3.5.5 I/O 信息	70
3.6 核心板快速选型表	76
第 4 章 ARM9、Cortex - A 无线核心板	78
4.1 A280 无线核心板 (ARM9 核)	78
4.1.1 概 述	78
4.1.2 产品特性	79
4.1.3 产品功能框图	80
4.1.4 产品选型	80
4.1.5 I/O 信息	81
4.2 A287 无线核心板 (ARM9 核)	87
4.2.1 概 述	87
4.2.2 产品特性	88
4.2.3 产品功能框图	89
4.2.4 产品选型	89
4.2.5 I/O 信息	90
4.3 A6G2C 无线核心板 (A7 核)	95
4.3.1 概 述	95
4.3.2 产品特性	95
4.3.3 产品功能框图	97
4.3.4 产品选型	97
4.3.5 I/O 信息	98
4.4 A3352 无线核心板 (A8 核)	103
4.4.1 概 述	103
4.4.2 产品特性	103
4.4.3 产品功能框图	104
4.4.4 产品选型	105
4.4.5 I/O 信息	105
4.5 无线核心板快速选型表	111
第 5 章 ARM9、Cortex - A 工控主板	114
5.1 i.MX28x 无线工控板 (ARM9 核)	114
5.1.1 概 述	114
5.1.2 产品特性	115
5.1.3 产品选型	116
5.2 IoT - 3968L 网络控制器 (ARM9 核)	116
5.2.1 概 述	116
5.2.2 产品特性	117



5.3 Cortex-A7 无线工控板(A7 核)	118
5.3.1 产品特性	119
5.3.2 产品选型	120
5.4 IoT7000A-LI 网络控制器(A7 核)	121
5.4.1 概述	121
5.4.2 产品特性	122
5.5 IoT-A3352LI 无线工控板(A8 核)	122
5.5.1 概述	122
5.5.2 产品特性	123
5.6 IoT3000A-AWI 网络控制器	124
5.6.1 概述	124
5.6.2 产品特性	125
5.7 工控主板快速选型表	126

第三部分 软件篇

第 6 章 通用设备接口	128
6.1 通用 LED 接口	128
6.2 通用键盘接口	131
6.3 通用蜂鸣器接口	137
6.4 通用数码管接口	141
6.5 通用传感器接口	146
6.6 温控器	159
第 7 章 通用外设接口	165
7.1 GPIO	165
7.1.1 I/O 配置	165
7.1.2 I/O 的申请和释放	168
7.1.3 普通 I/O 接口	170
7.1.4 中断 I/O 接口	174
7.2 PWM	178
7.2.1 PWM 简介	178
7.2.2 PWM 接口	179
7.3 SPI 总线	183
7.3.1 SPI 总线简介	183
7.3.2 SPI 总线接口	185
7.4 I ² C 总线	195
7.4.1 I ² C 总线简介	195
7.4.2 I ² C 接口	195

7.5	UART 总线	203
7.5.1	UART 简介	203
7.5.2	串行接口	205
7.6	A/D 转换器	210
7.6.1	模/数信号转换	210
7.6.2	A/D 转换接口	214
7.7	D/A 转换器	229
7.7.1	数/模信号转换	229
7.7.2	D/A 转换接口	230
7.8	看门狗(WDT)	234
7.8.1	看门狗简介	234
7.8.2	看门狗接口	237
第 8 章	时间管理	241
8.1	时间的表示形式	241
8.1.1	细分时间	241
8.1.2	日历时间	242
8.1.3	精确日历时间	243
8.1.4	细分时间与日历时间的相互转换	243
8.2	RTC 通用接口	245
8.3	系统时间	247
8.3.1	获取系统时间	247
8.3.2	设置系统时间	250
8.4	系统节拍	252
8.5	软件定时器	257
第 9 章	内存管理	262
9.1	堆管理器	262
9.1.1	堆管理器的原理概述	262
9.1.2	堆管理器接口	267
9.1.3	系统堆管理	273
9.2	内存池	279
9.2.1	内存池原理概述	280
9.2.2	内存池接口	281
第 10 章	实时内核	288
10.1	任务管理	288
10.1.1	多任务环境简介	288
10.1.2	创建任务	291



10.1.3	终止任务	298
10.1.4	任务延时	299
10.1.5	检查栈空间的使用情况	300
10.2	信号量	303
10.2.1	互斥信号量	304
10.2.2	二进制信号量	312
10.2.3	计数信号量	321
10.3	邮 箱	326
10.4	消息队列	339
10.5	自旋锁	347
第 11 章 文件系统		351
11.1	文件系统简介	351
11.2	设备挂载管理	352
11.3	文件基本操作	359
11.4	目录基本操作	372
11.5	微型数据库	378
11.5.1	哈希表	378
11.5.2	微型数据库接口	382
第 12 章 AWBus - lite 总线框架		392
12.1	AWBus - lite 简介	392
12.2	AWBus - lite 拓扑结构	394
12.3	系统硬件资源	395
12.3.1	硬件设备列表	396
12.3.2	设备描述类型	396
12.3.3	设备描述宏定义	403
12.3.4	设备的配置与裁剪	404
12.3.5	注册设备驱动	407
12.3.6	硬件设备的父总线设备	408
12.4	访问设备	409
12.4.1	通用接口	409
12.4.2	资源 ID	411
第 13 章 深入理解 AWBus - lite		413
13.1	通用接口的定义	413
13.1.1	接口命名	413
13.1.2	接口参数	414
13.1.3	返回值	414

13.2 接口的实现	415
13.2.1 实现接口初探	415
13.2.2 LED 抽象方法	417
13.2.3 抽象的 LED 服务	418
13.2.4 Method 机制	422
13.2.5 LED 服务链表的初始化	426
13.3 设备驱动	429
13.3.1 基础驱动信息	429
13.3.2 实际驱动信息	435
13.3.3 定义设备类型	436
13.3.4 定义设备信息类型	437
13.3.5 实现三个阶段的初始化函数	440
13.3.6 实现 LED 服务	443
13.3.7 定义 Method 对象	447
13.3.8 注册驱动	448
13.4 驱动开发的一般方法	450
13.4.1 定义驱动名	452
13.4.2 确定总线类型和设备类型	452
13.4.3 定义设备类型	452
13.4.4 定义设备信息类型	454
13.4.5 实现三个阶段的初始化函数	455
13.4.6 实现通用服务	459
13.4.7 定义 Method 对象	464
13.4.8 定义驱动结构体常量,实现驱动注册函数	465
第 14 章 信号采集及接口扩展模块	471
14.1 温度检测模块——TPS0xR/T	471
14.1.1 TPS02R 简介	472
14.1.2 添加 TPS02R 硬件设备	474
14.1.3 使用 TPS02R 模块	478
14.2 能效检测模块——EMM400x	479
14.2.1 EMM400A 简介	480
14.2.2 添加 EMM400 硬件设备	483
14.2.3 使用 EMM400 模块	486
14.3 信号调理模块——TPS0xU	488
14.3.1 TPS08U 简介	488
14.3.2 添加 TPS08U 硬件设备	490
14.3.3 使用 TPS08U 模块	494
14.4 接口扩展模块——RTM11AT	496



14.4.1	RS485 简介	496
14.4.2	RTM11AT 简介	497
14.4.3	添加 RTM11AT 硬件设备	500
14.4.4	使用 RTM11AT 模块	504
第 15 章 常用外围器件		506
15.1	EEPROM 存储器	506
15.1.1	器件简介	506
15.1.2	添加 EEPROM 硬件设备	507
15.1.3	NVRAM 通用接口	512
15.2	SPI NOR Flash 存储器	515
15.2.1	器件简介	515
15.2.2	添加 SPI NOR Flash 硬件设备	516
15.2.3	使用 MTD 存储器	524
15.2.4	使用 NVRAM 存储段	529
15.3	RTC 实时时钟	530
15.3.1	基于 RTC 通用接口的应用程序	531
15.3.2	RX8025T	532
15.3.3	DS1302	537
15.4	ZLG72128——数码管与键盘管理	542
15.4.1	ZLG72128 简介	542
15.4.2	添加 ZLG72128 硬件设备	546
15.4.3	使用 ZLG72128 数码管功能	552
15.4.4	使用 ZLG72128 按键功能	553
15.5	ADS131E0x——A/D 转换芯片	555
15.5.1	器件简介	556
15.5.2	添加 ADS131E08 硬件设备	559
15.5.3	使用 ADS131E08	564
15.6	AD5689R——D/A 转换芯片	571
15.6.1	器件简介	571
15.6.2	添加 AD5689R 硬件设备	574
15.6.3	使用 AD5689R	579
参 考 文 献		580

第一部分

简介篇

第 1 章

AWorks 简介

本章导读

经过十多年的不断研发、积累和完善,ZLG 推出了全新的 AWorks 平台——IoT 物联网生态系统,并已成功地应用到 ZLG 的示波器、功率计、功率分析仪、电压监测仪、电能质量分析仪、数据记录仪与工业通信等系列高性能仪器和工业 IoT 产品中。AWorks 的诞生将极大地降低开发者门槛,为开发者提供便利,使其可以忽略底层技术细节,专注产品“核心域”,更快地开发出具有竞争力的产品。同时,AWorks 为开发者提供的是高度抽象的通用接口,基于 AWorks 平台的软件与底层硬件无关,可以“随心所欲”的跨平台复用(如更换 MCU、外围器件等)。

AWorks 是在怎样的背景下诞生的? AWorks 究竟是什么? 怎样使用 AWorks? 本章将逐一解答这些疑问,使读者对 AWorks 有一个初步的认识。

1.1 诞生的背景

虽然嵌入式系统和通用计算机系统同源,但由于应用领域和研发人员不同,嵌入式系统很早就走向了相对独立的发展道路。通用计算机软件帮助人们解决了各种繁杂的问题,但随着需求的提升,所面临的问题也越来越复杂,软件领域的大师们对这些问题进行了深入研究和实践,于是诞生了科学的软件工程理论。无需多言,通用计算机软件的发展是我们有目共睹的。

再回过头来看嵌入式系统的发展,其需求相对来说较为简单,比如,通过热电阻传感器测温、上下限报警与继电器的动作等,因此嵌入式系统的应用开发似乎没有必要使用复杂的软件工程方法,从而使通用计算机系统和嵌入式系统走上了不同的发展道路。

嵌入式系统发展到今天,所面对的问题也日益变得复杂起来,而编程模式却没有多大的进步,这就是其所面临的困境。相信大家都或多或少地感觉到了,嵌入式系统行业的环境已经开始发生根本的改变,智能硬件和工业互联网等都让人始料不及,危机感油然而生。