

高等院校“十三五”精品规划教材

arm University Program

Arm 大学计划推荐用书

基于**ARM Cortex-M4** 内核的物联网/嵌入式系统 开发教程

主 编 刘 雯 副主编 姜铁增 陈 炜 雷 磊

企业支持

名校课程

专业前沿

面向应用



中国水利水电出版社
www.waterpub.com.cn

高等院校“十三五”精品规划教材

基于 ARM Cortex-M4 内核的 物联网/嵌入式系统开发教程

主 编 刘 雯

副主编 姜铁增 陈 炜 雷 磊



 中国水利水电出版社
www.waterpub.com.cn

· 北京 ·

内 容 提 要

本书主要内容包括：物联网技术的架构及应用；嵌入式系统的组成以及开发工具；Cortex-M4 内核；STM32F401 芯片的体系架构以及功能模块；基于 STM32F401 芯片的实例开发，包括 GPIO、中断机制、串口通信、AD 转换器、低功耗蓝牙、传感器模块、小型物联网系统和云服务系统最简模型等。

本书面向物联网开发的初学者和大专院校电子科学与技术及通信类专业的学生。全书贯穿物联网核心内容——感知、通信、信息处理、端到云的拓展等组成部分，以应用最广的基于 ARM 的经典嵌入式设备为载体，结合应用需求，用浅显易懂的语言以及各种实例对嵌入式物联网开发的知识进行系统讲解，使读者快速上手，并且为以后的物联网开发打下坚实的基础。

图书在版编目（C I P）数据

基于ARM Cortex-M4内核的物联网/嵌入式系统开发教程 / 刘雯主编. — 北京：中国水利水电出版社，2018.1（2018.9 重印）
高等院校“十三五”精品规划教材
ISBN 978-7-5170-6275-2

I. ①基… II. ①刘… III. ①互联网络—应用—高等学校—教材②智能技术—应用—高等学校—教材③网络编程—高等学校—教材 IV. ①TP393.409②TP18③TP311.1

中国版本图书馆CIP数据核字(2018)第012663号

策划编辑：杨庆川 责任编辑：周益丹 加工编辑：高双春 封面设计：李 佳

书 名	高等院校“十三五”精品规划教材 基于 ARM Cortex-M4 内核的物联网/嵌入式系统开发教程 JIYU ARM Cortex-M4 NEIHE DE WULIANWANG / QIANRUSHI XITONG KAIFA JIAOCHENG
作 者	主 编 刘 雯 副主编 姜铁增 陈 炜 雷 磊
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路 1 号 D 座 100038） 网址：www.waterpub.com.cn E-mail: mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）68367658（营销中心）、82562819（万水）
经 售	全国各地新华书店和相关出版物销售网点
排 版	北京万水电子信息有限公司
印 刷	三河市鑫金马印装有限公司
规 格	184mm×260mm 16 开本 15 印张 370 千字
版 次	2018 年 1 月第 1 版 2018 年 9 月第 2 次印刷
印 数	3001—6000 册
定 价	39.00 元

凡购买我社图书，如有缺页、倒页、脱页的，本社营销中心负责调换
版权所有·侵权必究

前 言

物联网技术（IoT）是新一代信息技术的重要组成部分，它将物理设备（传感器设备）与网络系统连接起来，并允许信息的全球共享。物联网在智能交通、智能电网、智能家居、智慧医疗以及工业和农业等多领域都有着广泛的应用，它与我们的生活越来越密不可分。随着云计算的发展，云服务在各个行业的应用迅速增加。物联网服务从端到云的拓展，大大提升了服务的可用性和可达性。同时，云端可以提供更多的已被设计完善、可共享或可定制的服务项目或引擎。对开发人员来说，学习物联网技术便是迫切的需求。嵌入式控制技术是物联网技术中重要的一环，而当前嵌入式开发相关教材众多，与物联网应用以及云服务紧密联系的教材有限，初学者想进行相应的学习在选择上有一定困难。

本书依托北京邮电大学电子工程学院开设的基于 ARM 的物联网应用实验课程，力图在书中兼顾嵌入式开发的核心内容及基于物联网架构的典型应用实例，希望读者可以通过本书的实训实例，在掌握物联网的基本架构的同时，利用分立或集成的传感器、嵌入式设备等硬件，能按照自己身边的实际需求，搭建不同的物联网应用原型。

本书的主要篇幅放在 Cortex-M4 内核、STM32F401 芯片的体系架构以及相关功能模块和基于 STM32F401 芯片的实例开发上。其中实例部分包括 GPIO、中断机制、串行通信、AD 转换器、低功耗蓝牙、传感器模块、小型物联网系统和简单的云服务系统等，读者可以逐步了解到一个物联网架构从局部到整体的搭建过程。如果对相应基础知识有了深刻的认识，读者也可跳过相应章节进行第 5 章以后相应模块的学习。

本书相关课程的开发得到了北京邮电大学电子工程学院提供的实践教学课程开发环境支持，感谢实验中心赵同刚主任和饶岚老师的支持。在本书编写过程中，ARM 中国大学计划负责人陈炜博士在内容及章节安排上提供了建设性的意见，并在代码编写以及其他资料上给予了极大的帮助；ARM 公司雷磊工程师在实验课程开发过程中提供了技术支持，他也作为副主编参与了物联网课程端到云部分的编撰工作；王梦馨助理提供了资料查询的帮助；同时感谢本书编辑杨庆川社长的大力支持。研究生莫耀凯、綦航、隋钰童、程倩倩和郑心雨等同学参与完成了的资料查找、文献翻译、代码调试以及校对工作。需要指出的是，本书的编写参考了大量的同类型教材以及相关技术论坛资料。在此对所有提到的单位和个人表示感谢。

物联网技术发展迅速，加之时间仓促，本书难免存在缺漏和错误，恳请同行及广大读者批评指正。提出书中的问题以及索要相关实例与 PPT 文件请发邮件至 qrsww@163.com。

编 者
2017 年 11 月

目 录

前言

第1章 嵌入式物联网开发绪论..... 1

1.1 物联网的基本概念..... 1

1.1.1 物联网的定义与特征..... 1

1.1.2 物联网的应用..... 1

1.2 物联网的体系架构..... 2

1.2.1 感知层..... 2

1.2.2 网络层..... 3

1.2.3 应用层..... 4

1.3 嵌入式系统..... 4

1.3.1 嵌入式系统简介..... 4

1.3.2 嵌入式系统的组成..... 5

1.3.3 物联网与嵌入式系统的关系..... 8

1.3.4 嵌入式系统开发软件——Keil..... 9

1.3.5 物联网设备开发与 Mbed..... 10

参考资料..... 12

第2章 ARM Cortex-M4 技术..... 13

2.1 背景概述..... 13

2.2 核心技术..... 14

2.2.1 内部架构..... 14

2.2.2 内核比较..... 15

2.2.3 Thumb-2 指令集..... 17

2.2.4 流水线技术..... 18

2.2.5 寄存器..... 19

2.2.6 工作模式..... 21

2.2.7 异常..... 22

参考资料..... 23

第3章 STM32F401 体系结构..... 25

3.1 STM32F401 架构..... 25

3.1.1 片内结构..... 25

3.1.2 功能单元描述..... 25

3.2 封装与引脚说明..... 28

参考资料..... 32

第4章 STM32F401 功能模块设计..... 34

4.1 电源模块..... 34

4.1.1 电源..... 34

4.1.2 电源监控器..... 35

4.1.3 低功耗模式..... 37

4.1.4 电源控制寄存器..... 38

4.2 复位模块..... 43

4.2.1 系统复位..... 43

4.2.2 电源复位..... 44

4.2.3 备份域复位..... 44

4.3 时钟管理模块..... 44

4.3.1 HSE 时钟..... 46

4.3.2 HSI 时钟..... 46

4.3.3 PLL 配置..... 47

4.3.4 LSE 时钟..... 47

4.3.5 LSI 时钟..... 47

4.3.6 系统时钟 (SYSCLK) 选择..... 48

4.4 定时器与看门狗..... 48

4.4.1 高级控制定时器 (TIM1)..... 48

4.4.2 通用定时器 (TIMx)..... 49

4.4.3 独立看门狗 (IWDG)..... 49

4.4.4 窗口看门狗 (WWDG)..... 50

4.4.5 SysTick 定时器..... 52

4.5 内部存储器模块..... 52

4.5.1 STM32F401 内部存储空间..... 52

4.5.2 Flash 存储器..... 52

4.5.3 RAM 数据存储器..... 55

参考资料..... 56

第5章 通用 I/O 接口..... 57

5.1 通用 I/O 功能描述..... 57

5.1.1 GPIO 端口..... 57

5.1.2 输入输出多路复用器和映射..... 57

5.1.3 I/O 端口寄存器..... 58

5.1.4 GPIO 模式..... 59

5.2 通用 I/O 配置寄存器..... 60

5.2.1 GPIO 端口模式寄存器

(GPIOx_MODER)	60
5.2.2 GPIO 输出类型寄存器	
(GPIOx_OTYPER)	60
5.2.3 GPIO 端口输出速度寄存器	
(GPIOx_OSPEEDR)	61
5.2.4 GPIO 端口上拉下拉寄存器	
(GPIOx_PUPDR)	61
5.2.5 GPIO 端口输入数据寄存器	
(GPIOx_IDR)	62
5.2.6 GPIO 端口输出数据寄存器	
(GPIOx_ODR)	62
5.2.7 GPIO 端口比特置位复位寄存器	
(GPIOx_BSRR)	63
5.2.8 GPIO 端口配置锁存器	
(GPIOx_LCKR)	63
5.2.9 GPIO 复用功能低位寄存器	
(GPIOx_AFR1)	64
5.2.10 GPIO 复用功能高位寄存器	
(GPIOx_AFR2)	65
5.2.11 RCC AHB1 外设时钟使能寄存器	
(RCC_AHB1ENR)	66
5.3 应用实例	66
5.3.1 开发环境与实例说明	66
5.3.2 Keil 软件使用	68
5.3.3 寄存器操作技巧	71
5.3.4 GPIO 实例代码	71
5.3.5 测试结果及分析	73
参考资料	73
第 6 章 STM32F401 中断机制	74
6.1 中断控制	74
6.1.1 基本概念	74
6.1.2 中断优先级	74
6.1.3 中断控制位	75
6.1.4 中断过程	75
6.1.5 外部中断/事件控制器 (EXTI)	76
6.1.6 外部中断/事件线映射	77
6.2 中断控制寄存器	78
6.2.1 NVIC 寄存器	78
6.2.2 EXTI 寄存器	79

6.3 应用实例	81
6.3.1 开发环境与实例说明	81
6.3.2 中断实例代码	82
6.3.3 测试结果及分析	84
参考资料	85
第 7 章 STM32F401 串行通信	86
7.1 USART 简介及主要功能	86
7.2 USART 功能描述	87
7.2.1 USART 结构	87
7.2.2 USART 字符描述	87
7.2.3 发送器	89
7.2.4 接收器	92
7.2.5 多处理器通信	95
7.2.6 LIN (局域互连网络) 模式	96
7.2.7 USART 同步模式	97
7.2.8 单线半双工通信	98
7.3 应用实例	98
7.3.1 开发环境与实例说明	98
7.3.2 UART 实例代码	99
7.3.3 测试结果及分析	103
参考资料	104
第 8 章 STM32F401 AD 转换器	105
8.1 功能描述	105
8.1.1 ADC 介绍	105
8.1.2 ADC 功能描述	105
8.2 ADC 寄存器配置	112
8.2.1 ADC 状态寄存器 (ADC_SR)	112
8.2.2 ADC 控制寄存器 1 (ADC_CR1)	114
8.2.3 ADC 寄存器 2 (ADC_CR2)	117
8.2.4 ADC 采样时间寄存器 1	
(ADC_SMPR1)	120
8.2.5 ADC 采样时间寄存器 2	
(ADC_SMPR2)	121
8.2.6 ADC 注入通道数据偏移寄存器	
(ADC_JOFRx) (x=1..4)	122
8.2.7 ADC 看门狗高阈值寄存器	
(ADC_HTR)	122
8.2.8 ADC 看门狗低阈值寄存器	
(ADC_LTR)	123

8.2.9	ADC 规则序列寄存器 1 (ADC_SQR1)	123	10.2.2	传感器的断开	157
8.2.10	ADC 规则序列寄存器 2 (ADC_SQR2)	124	10.2.3	可能用到的函数	157
8.2.11	ADC 规则序列寄存器 3 (ADC_SQR3)	124	10.2.4	程序框架	159
8.2.12	ADC 注入序列寄存器 (ADC_JSQR)	125	10.3	应用实例	159
8.2.13	ADC 注入数据寄存器 x (ADC_JDRx) (x=1..4)	126	10.3.1	开发环境与实例说明	159
8.2.14	ADC 规则数据寄存器 (ADC_DR)	126	10.3.2	传感器模块实例代码	160
8.2.15	ADC 通用控制寄存器 (ADC_CCR)	126	10.3.3	测试结果及分析	161
8.2.16	ADC 寄存器映射	127		参考资料	161
8.3	应用实例	128	第 11 章	嵌入式物联网系统设计与实例	163
8.3.1	开发环境与实例说明	128	11.1	传感器数据采集	163
8.3.2	实例代码	128	11.1.1	温度传感器	163
8.3.3	测试结果及分析	132	11.1.2	温湿度传感器	163
	参考资料	132	11.1.3	超声波传感器	164
第 9 章	STM32F401 低功耗蓝牙	133	11.1.4	烟雾传感器	164
9.1	功能描述	133	11.1.5	声音传感器	165
9.1.1	蓝牙技术简介	134	11.1.6	光敏传感器	165
9.1.2	BlueNRG	142	11.2	蓝牙气象站实例	165
9.1.3	BALF-NRG-01D3	142	11.2.1	开发环境与实例说明	166
9.2	蓝牙模块配置	143	11.2.2	蓝牙气象站实例代码	166
9.2.1	Mbed	143	11.2.3	测试结果及分析	177
9.2.2	可能用到的函数	144	11.3	设计建议	177
9.2.3	程序框架	145	第 12 章	物联网和云	180
9.3	应用实例	146	12.1	物联网需要云	180
9.3.1	开发环境与实例说明	146	12.1.1	云计算	180
9.3.2	蓝牙实例代码	147	12.1.2	云计算的基本概念术语	180
9.3.3	测试结果及分析	149	12.1.3	云计算的安全	188
	参考资料	149	12.2	物联网与云的结合	189
第 10 章	STM32F401 传感器模块	151	12.2.1	物联网的端到云	189
10.1	功能描述	151	12.2.2	物联网与云计算结合的模式分类	190
10.2	传感器模块配置	157	12.2.3	物联网与云计算的分阶段融合	191
10.2.1	传感器 I ² C 地址的选择	157	12.2.4	物联网与云计算的结合优势	191
			12.2.5	物联网与云的结合实例	193
			12.3	使用 Bluemix 连接设备实例	195
			12.3.1	开发环境与实例说明	196
			12.3.2	测试结果及分析	202
				参考资料	203
			附录 1	Keil 软件使用详细教程	205
			附录 2	Mbed 编程实例代码	217

第1章 嵌入式物联网开发绪论

1.1 物联网的基本概念

1.1.1 物联网的定义与特征

物联网 (Internet of Things, IoT) 是新一代信息技术的重要组成部分, 也是“信息化”时代的重要发展阶段。其定义为: 通过射频识别 (RFID)、红外感应器、全球定位系统、激光扫描器等信息传感设备, 按约定的协议把任何物品与互联网连接起来, 进行信息交换和通信, 以实现智能化识别、定位、跟踪、监控和管理的一种网络。

由于物联网是通过各种感知设备和互联网连接物体与物体, 实现全自动、智能化采集, 传输与处理信息, 达到随时随地进行科学管理目的的一种网络, 所以, “感知化”“互联化”“智能化”是物联网的基本特征。

(1) 感知化: 物联网是各种感知技术的广泛应用。物联网上部署了海量的不同类型的传感器, 包括射频识别 (RFID) 装置、红外感应器、全球定位系统、激光扫描器等信息传感设备, 它们是物联网不可或缺的关键元器件。有了它们才可以实现近 (远) 距离、无接触、自动化感应和数据读出、数据发送等。

(2) 互联化: 物联网是一种建立在互联网上的泛在网络。物联网技术的重要基础和核心仍旧是互联网, 通过各种有线和无线网络与互联网融合, 将物体的信息实时准确地传递出去。在物联网上的传感器定时采集的信息需要通过网络传输, 由于其数量极其庞大, 形成了海量信息, 在传输过程中, 为了保障数据的正确性和及时性, 必须适应各种异构网络和协议。

(3) 智能化: 物联网不仅仅提供了传感器的连接, 其本身也具有智能处理的能力, 能够对物体实施智能控制。物联网将传感器和智能处理相结合, 利用云计算、模式识别等各种智能技术, 扩充其应用领域。从传感器获得的海量信息中分析、加工和处理出有意义的信息, 以适应不同用户的需求, 发现新的应用领域和应用模式。

1.1.2 物联网的应用

物联网用途广泛, 遍及智能交通、智能家居、智慧医疗、智能电网、智能工业、智慧农业、智能物流、智能安保等多个领域。本节重点讲述智能交通、智能电网、智能家居、智慧医疗四个方面。

1. 智能交通

智能交通系统包括电子收费系统、实时交通信息服务、智能交通管理等应用。这里简单介绍电子收费系统和实时交通信息服务。

在电子收费系统中, 车辆需安装一个系统可唯一识别的称为电子标签的设备, 且在收费站的道口或公路上设置可读/写该电子标签的标签读写器和相应的计算机收费系统。车辆通过

收费站时,驾驶员不必停车交费,只需以系统允许的速度通过,车载电子标签便可自动与安装在路侧或门架上的标签读写器进行信息交换,收费计算机收集通过车辆信息,并将收集到的信息上传给后台服务器,服务器根据这些信息识别出道路使用者,然后自动从道路使用者的账户中扣除通行费。实时交通信息服务是一种协同感知类任务,设置在各交通路口的传感器实时感知路况信息,并实时上传到主控中心,经过数据挖掘与交通规划分析系统,对海量信息进行数据融合和分析处理,并经通信塔发布给市民。

2. 智能电网

智能电网是以双向数字科技创建的输电网络,用来传送电力。它可以侦测电力供应者的电力供应状况和一般家庭用户的电力使用状况,从而调整家用电器的耗电量,以此达到节约能源、降低损耗、增强电网可靠性的目的。具体而言,感知控制子层主要通过各种新型 MEMS 传感器、基于嵌入式系统的智能传感器、智能采集设备等技术手段,实现对物质属性、环境状态、行为态势等静态或动态的信息进行大规模、分布式的信息获取。网络层以电力光纤网为主,以电力线载波通信网、无线宽带网为辅,将感知层设备采集数据进行转发,负责物联网与智能电网专用通信网络之间的接入,主要用来实现信息的传递、路由和控制。面向智能电网的应用通过运用智能计算、模式识别等技术来实现电网相关数据信息的整合分析处理,进而实现智能化的决策、控制和服务。

3. 智能家居

智能家居产品融合自动化控制系统、计算机网络系统和网络通信技术于一体,将各种家庭设备(如音视频设备、照明系统、窗帘控制、空调控制、安防系统、数字影院系统、网络家电等)通过智能家庭网络实现自动化。用户通过运营商的宽带、固定电话和无线网络,可以实现对家庭设备的远程操控。与普通家居相比,智能家居不仅能提供舒适宜人且高品位的家庭生活空间,实现更智能的家庭安防系统,还将家居环境由原来的被动静止结构转变为具有能动智慧的工具,提供全方位的信息交互功能。

4. 智慧医疗

智慧医疗系统借助简单实用的家庭医疗传感设备,对家中病人或老人的生理指标进行检测,并将生成的生理指标数据通过固定网络或 4G 无线网络传送给护理人或有关医疗单位。根据客户的需求,信息服务商还提供相关增值业务,如紧急呼叫救助服务、专家咨询服务、终生健康档案管理服务等。

1.2 物联网的体系架构

物联网作为一种形式多样的聚合性复杂系统,涉及了信息技术自上而下的每一个层面,其体系架构一般可分为感知层、网络层、应用层 3 个层面,如图 1-1 所示。其中,公共技术不属于物联网技术的某个特定层面,而是与物联网技术架构的 3 层都有关系,它包括标识与解析、安全技术、网络管理和服务质量(QoS)管理等内容。

1.2.1 感知层

感知层主要解决对物理世界数据获取的问题,从而达到对数据全面感知的目的。它由数据采集子层、短距离通信技术和协同信息处理子层组成。数据采集子层通过各种类型的传感器

获取物理世界中发生的物理事件和数据信息，例如，各种物理量、标识、音视频多媒体数据。物联网的数据采集涉及传感器、RFID、多媒体信息采集、二维码和实时定位等技术。短距离通信技术和协同信息处理子层将采集到的数据在局部范围内进行协同处理，以提高信息的精准度，降低信息冗余度，并通过具有自组织能力的短距离传感网接入广域承载网络。感知层处于物联网三层架构的最底层，是物联网中最基础的连接与管理对象。在物联网中，各类感知装置不仅要解决“上行”的感知与检测问题，而且要实现“下行”的监测与控制问题，达到“监、督、控”的一体化。

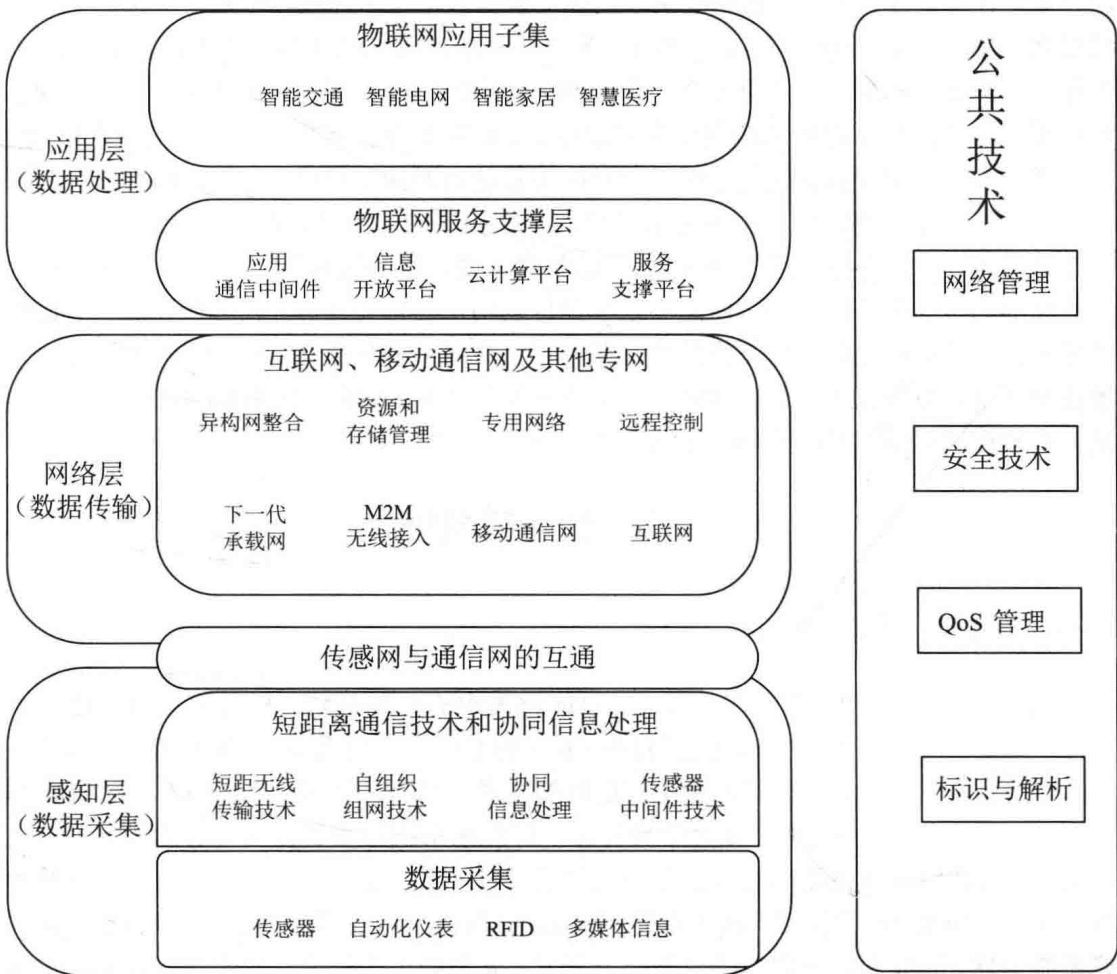


图 1-1 物联网的体系架构

1.2.2 网络层

网络层主要实现信息的传递、路由和控制，以及将来自感知层的各类信息通过基础承载网络传输到应用层，包括移动通信网、互联网、卫星网、广电网、行业专网，及形成的融合网络等。根据应用需求，可作为透传的网络层，也可升级以满足未来不同内容传输的要求。网络层主要关注来自于感知层的、经过初步处理的数据经由各类网络的传输问题。这涉及智能路由器，不同网络传输协议的互通、自组织通信等多种网络技术。

1.2.3 应用层

应用层的任务是利用云计算、模糊识别等智能计算技术,将各类物联网的服务以用户需要的形式呈现出来,达到信息最终为人所用的目的。应用层主要包括物联网服务支撑层和物联网应用子集层。物联网的核心功能是对信息资源进行采集、开发和利用,因此这部分内容十分重要。服务支撑层的主要功能是根据底层采集的数据,形成与业务需求相适应、实时更新的动态数据资源库。该部分将采用元数据注册、发现元数据、信息资源目录、互操作元模型、分类编码、并行计算、数据挖掘、数据收割、智能搜索等各项技术开展物联网数据体系结构、信息资源规划、信息资源库设计和维护等技术;各个业务场景可以在此基础上,根据业务需求特点,开展相应的数据资源管理。应用层将为各类业务提供统一的信息资源支撑,通过建立并实时更新和重组使用的信息资源库和应用服务资源库,使得各类业务服务根据用户的需求组合,对于业务的适应能力明显提高。该层能够提升对应用系统资源的重用度,为快速构建新的物联网应用奠定基础,满足在物联网环境中复杂多变的网络资源应用需求和服务。

除此之外,物联网还需要信息安全、物联网管理、服务质量管理等公共技术支撑,以采用现有标准为主。在各层之间,信息不是单向传递的,是有交互、控制等操作的,所传递的信息多种多样,其中最为关键的是围绕物品信息,完成海量数据采集、标识解析、传输、智能处理等各个环节,与各业务领域应用融合,完成各业务功能。因此,物联网的系统架构和标准体系是一个紧密关联的整体,引领了物联网领域研究的方向。

1.3 嵌入式系统

1.3.1 嵌入式系统简介

嵌入式系统是以计算机技术为基础,以应用为中心,并且软、硬件可裁剪,适用于应用系统对功能、可靠性、功耗、成本等有严格要求的专用计算机系统。IEEE 定义嵌入式系统是“控制、监视或者辅助设备、机器和工厂运行的装置”。嵌入式系统包括 CPU、存储器和输入/输出设备三部分。其核心由一个或几个预先编程好以用来执行少数几项任务的微处理器或者单片机组成。与通用计算机能够运行的用户选择的软件不同,嵌入式系统上的软件通常是暂时不变的,所以经常称为“固件”。其主要特点包括系统的可靠性、稳定性高,容错能力良好;嵌入式系统的软、硬件面向特定任务对象进行开发;大多嵌入式系统对于实时性及系统快速反应能力有很高要求;具有丰富的外设接口及友好的交互界面等。嵌入式系统从早期的军事航天领域开始发展,逐步广泛应用于汽车电子、通信、消费电子、工业控制等领域。随着医疗电子、智能家居、物流管理和电力控制等方面的不断风靡,嵌入式系统利用自身积累的底蕴经验,重视和把握这个机会,在已经成熟的平台和产品基础上与应用传感单元结合,扩展物联和感知的支持能力,发掘某种领域的物联网应用。

嵌入式系统的主要研究内容如下:

- (1) VHDL/Verilog 硬件描述语言; FPGA/CPLD 固件载体; 相应 EDA 工具。
- (2) IP Core 与基于 IP Core 的 SoC/SoPC 芯片级设计。
- (3) EMPU/EMC/DSP 与基于平台的嵌入式系统设计。

- (4) CPU 硬核（硬微处理机）与固核（固微处理器）。
- (5) RTOS 的移植与裁减。
- (6) 嵌入式系统软/硬件协同设计。
- (7) 嵌入式系统低功耗设计。
- (8) 嵌入式 Internet 系统。
- (9) 关键技术：USB、TCP/IP、FAT、GUI。

1.3.2 嵌入式系统的组成

一个嵌入式系统装置一般都由嵌入式计算机系统和执行装置组成，如图 1-2 所示，其中，嵌入式计算机系统是整个嵌入式系统的核心，由硬件层、中间层、系统软件层和应用软件层组成。执行装置也称为被控对象，它可以接受嵌入式计算机系统发出的控制命令，执行所规定的操作或任务。

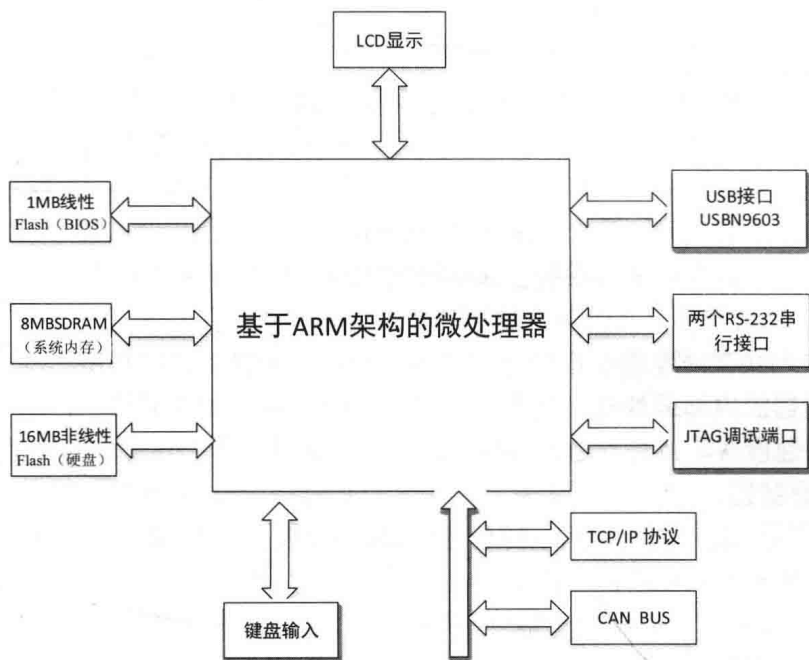


图 1-2 ARM 架构图

1. 硬件层

硬件层中包含嵌入式微处理器、存储器（SDRAM、ROM、Flash 等）、通用设备接口和 I/O 接口（A/D、D/A、I/O 等）。在一片嵌入式处理器基础上添加电源电路、时钟电路和存储器电路，就构成了一个嵌入式核心控制模块。其中操作系统和应用程序都可以固化在 ROM 中。本书介绍的嵌入式系统就是 STM32F401 开发板。

(1) 嵌入式微处理器。

嵌入式系统硬件层的核心是嵌入式微处理器，嵌入式微处理器与通用 CPU 最大的不同在于嵌入式微处理器大多工作在为特定用户群所专门设计的系统中，它将通用 CPU 中许多由板卡完成的任务集成在芯片内部，从而有利于嵌入式系统在设计时趋于小型化，同时还具有很高

的效率和可靠性。

嵌入式微处理器的体系结构可以采用冯·诺依曼体系或哈佛体系结构；指令系统可以选择精简指令系统(Reduced Instruction Set Computer, RISC)和复杂指令系统(Complex Instruction Set Computer, CISC)。RISC 计算机在通道中只包含最有用的指令，确保数据通道快速执行每一条指令，从而提高了执行效率并使 CPU 硬件结构设计变得更为简单。

嵌入式微处理器有各种不同的体系，即使在同一体系中也可能具有不同的时钟频率和数据总线宽度，或集成了不同的外设和接口。据不完全统计，目前全世界嵌入式微处理器已经超过 1000 多种，体系结构有 30 多个系列，其中主流的体系有 ARM、MIPS、PowerPC、X86 和 SH 等。但与全球 PC 市场不同的是，没有一种嵌入式微处理器可以主导市场，仅以 32 位的产品而言，就有 100 种以上的嵌入式微处理器。嵌入式微处理器的选择是根据具体的应用而决定的。

(2) 存储器。

嵌入式系统需要存储器来存放和执行代码。嵌入式系统的存储器包含 Cache、主存和辅助存储器。

1) Cache。

Cache 是一种容量小、速度快的存储器阵列，它位于主存和嵌入式微处理器内核之间，存放的是最近一段时间微处理器使用最多的程序代码和数据。在需要进行数据读取操作时，微处理器尽可能地从 Cache 中读取数据，而不是从主存中读取，这样就大大改善了系统的性能，提高了微处理器和主存之间的数据传输速率。Cache 的主要目标就是：减小存储器（如主存和辅助存储器）给微处理器内核造成的存储器访问瓶颈，使处理速度更快，实时性更强。

2) 主存。

主存是嵌入式微处理器能直接访问的寄存器，用来存放系统和用户的程序及数据。它可以位于微处理器的内部或外部，其容量为 256KB~1GB，根据具体的应用而定，一般片内存储器容量小，速度快，片外存储器容量大。

3) 辅助存储器。

辅助存储器用来存放大数据量的程序代码或信息，它的容量大、但读取速度与主存相比就慢很多，用来长期保存用户的信息。

(3) 通用设备接口和 I/O 接口。

嵌入式系统和外界交互需要一定形式的通用设备接口，如 A/D、D/A、I/O 等，外设通过和片外其他设备或传感器的连接来实现微处理器的输入/输出功能。每个外设通常都只有单一的功能，它可以在芯片外，也可以内置在芯片中。外设的种类很多，可从一个简单的串行通信设备到非常复杂的 802.11 无线设备。

目前嵌入式系统中常用的通用设备接口有 A/D（模/数转换接口）、D/A（数/模转换接口）、I/O 接口有 RS-232 接口（串行通信接口）、Ethernet（以太网接口）、USB（通用串行总线接口）、音频接口、VGA 视频输出接口、I²C（现场总线）、SPI（串行外围设备接口）和 IrDA（红外线接口）等。

2. 中间层

硬件层与软件层之间为中间层，也称为硬件抽象层（Hardware Abstract Layer, HAL）或板级支持包（Board Support Package, BSP），它将系统上层软件与底层硬件分离开来，使系统的底层驱动程序与硬件无关，上层软件开发人员无需关心底层硬件的具体情况，根据 BSP 层

提供的接口即可进行开发。该层一般包含相关底层硬件的初始化、数据的输入/输出操作和硬件设备的配置功能。

实际上, **BSP** 是一个介于操作系统和底层硬件之间的软件层次, 包括了系统中大部分与硬件联系紧密的软件模块。设计一个完整的 **BSP** 需要完成两部分工作: ①嵌入式系统的硬件初始化以及 **BSP** 功能; ②设计硬件相关的设备驱动。

(1) 嵌入式系统硬件初始化。

系统初始化过程可以分为 3 个主要环节, 按照自底向上、从硬件到软件的次序依次为: 片级初始化、板级初始化和系统级初始化。

1) 片级初始化。

完成嵌入式微处理器的初始化, 包括设置嵌入式微处理器的核心寄存器和控制寄存器、嵌入式微处理器核心工作模式和嵌入式微处理器的局部总线模式等。片级初始化把嵌入式微处理器从上电时的默认状态逐步设置成系统所要求的工作状态。这是一个纯硬件的初始化过程。

2) 板级初始化。

完成嵌入式微处理器以外的其他硬件设备的初始化。另外, 还需设置某些软件的数据结构和参数, 为随后的系统级初始化和应用程序的运行建立硬件和软件环境。这是一个同时包含软硬件两部分在内的初始化过程。

3) 系统初始化。

该初始化过程以软件初始化为主, 主要进行操作系统的初始化。**BSP** 将对嵌入式微处理器的控制权转交给嵌入式操作系统, 由操作系统完成余下的初始化操作, 包含加载和初始化与硬件无关的设备驱动程序, 建立系统内存区, 加载并初始化其他系统软件模块, 如网络系统、文件系统等。最后, 操作系统创建应用程序环境, 并将控制权交给应用程序的入口。

(2) 硬件相关的设备驱动程序。

BSP 的另一个主要功能是硬件相关的设备驱动。硬件相关的设备驱动程序的初始化通常是一个从高到低的过程。尽管 **BSP** 中包含硬件相关的设备驱动程序, 但是这些设备驱动程序通常不直接由 **BSP** 使用, 而是在系统初始化过程中由 **BSP** 将他们与操作系统中通用的设备驱动程序关联起来, 并在随后的应用中由通用的设备驱动程序调用, 实现对硬件设备的操作。与硬件相关的驱动程序是 **BSP** 设计与开发中另一个非常关键的环节。

3. 系统软件层

系统软件层由实时多任务操作系统 (**Real-time Operation System, RTOS**)、文件系统、图形用户接口 (**Graphic User Interface, GUI**)、网络系统及通用组件模块组成。**RTOS** 是嵌入式应用软件的基础和开发平台。

(1) 嵌入式操作系统。

嵌入式操作系统 (**Embedded Operation System, EOS**) 是一种用途广泛的系统软件, 过去它主要应用于工业控制和国防系统领域。**EOS** 负责嵌入系统的全部软、硬件资源的分配和任务调度, 控制、协调并发活动。它必须体现其所在系统的特征, 能够通过装卸某些模块来达到系统所要求的功能。目前, 已推出一些应用比较成功的 **EOS** 产品系列。随着 **Internet** 技术的发展、信息家电的普及应用及 **EOS** 的微型化和专业化, **EOS** 开始从单一的弱功能向高专业化的强功能方向发展。嵌入式操作系统在系统实时高效性、硬件的相关依赖性、软件固化以及应用的专用性等方面具有较为突出的特点。

（2）实时操作系统。

实时操作系统（Real Time Operating System，RTOS）是指具有实时性，能支持实时控制系统工作的操作系统。实时操作系统的首要任务是调动一切可利用的资源完成实时控制任务，其次才着眼于提高计算机系统的工作效率，其重要特点是通过任务调度来满足对于重要事件在规定时间内作出正确的响应。

实时操作系统是嵌入式应用软件的基础和开发平台，也是一段嵌入在目标代码中的软件，用户的其他应用程序都建立在 RTOS 之上。不但如此，RTOS 还是一个可靠性和可信性很高的实时内核，将 CPU 时间、中断、I/O、定时器等资源都包装起来，留给用户一个标准的 API，并根据各个任务的优先级，合理地在不同任务之间分配 CPU 时间。

1.3.3 物联网与嵌入式系统的关系

首先，在技术层面上，物联网与嵌入式系统都是多学科、多种技术融合的综合性的应用技术，物联网技术包含了嵌入式系统技术，物联网的发展需要嵌入式系统的支持。其次，在物联网的“物”与嵌入式系统关系层面上，物联网的“物”需要用到复杂的、网络化的、智能化的嵌入式系统。而在应用领域方面，两者几乎是相同的，当前物联网涉足的领域，嵌入式系统都已经在其中被使用了。综上所述，物联网与嵌入式系统关系非常紧密，物联网的发展离不开嵌入式系统的支持，而物联网又给嵌入式系统带来了新的发展机遇和挑战。

简单讲，物联网是物与物、人与物之间的信息传递与控制；专业上讲，物联网则是智能终端的网络化。嵌入式系统无所不在，有嵌入式系统的地方才会有物联网的应用。从另一个意义上也可以说，物联网的产生是嵌入式系统高速发展的必然产物，更多的嵌入式智能终端产品有了联网的需求，催生了物联网这个概念的产生。

从两者的定义来看，物联网强调的是物联网中设备具有感知、计算、执行、协同工作和通信能力及能提供的服务；嵌入式系统强调的是嵌入到宿主对象的专用计算系统，其功能或能提供的服务也比较单一。嵌入式系统具有的功能和物联网设备中某个子功能类似。因此，嵌入式系统被广泛应用于物联网各层的设备中。简单的嵌入式系统与物联网定义中的设备或者物有较大的区别，具有的功能不如物联网中的设备。但是随着嵌入式系统的不断发展，目前出现的一些复杂的嵌入式系统（如智能移动电话）基本上达到了物联网定义中的设备要求。从技术角度来看，首先，物联网与嵌入式系统都是各种技术融合的综合性技术，融合的技术大致相同；其次，物联网技术中又包含有嵌入式系统技术，如表 1-1 所示。

表 1-1 各种技术与物联网嵌入式系统的关系

技术	物联网	嵌入式系统
射频识别技术	需要	可选
电子技术	必需	必需
传感器技术	需要	必需
半导体技术	必需	必需
通信技术	必需	可选
智能计算技术	必需	可选

续表

技术	物联网	嵌入式系统
自动控制技术	可选	可选
软件技术	必需	必需

物联网的应用领域相当广泛，如航空航天、汽车工业、通信业、智能建筑、医药与医疗设备、交通运输、零售物流与供应链管理、农业种植、多媒体与娱乐、节能环保、环境监测等。经过近 20 年的快速发展，嵌入式系统已经从控制设备输入、输出的简单应用扩展到了影响人们生产生活的各个领域之中。嵌入式系统的行业应用有办公设备、建筑物设计、制造和流程设计、医疗、监视、卫生设备、交通运输、通信等。比较嵌入式系统与物联网的应用领域，可以发现它们几乎是一致的。这也说明了嵌入式系统是物联网发展的基础，物联网是嵌入式系统发展到一定阶段的产物。

1.3.4 嵌入式系统开发软件——Keil

1. Keil C51

Keil C51 是美国 Keil Software 公司出品的 51 系列兼容单片机 C 语言软件开发系统，与汇编相比，C 语言在功能性、结构性、可读性、可维护性上有明显的优势，因而易学易用。Keil 提供了包括 C 语言编译器、宏汇编、链接器、库管理和一个功能强大的仿真调试器等在内的完整开发方案，通过一个集成开发环境（ μ Vision）将这些部分组合在一起。

C51 工具包的整体结构， μ Vision 与 Ishell 分别是用于 Windows 和用于 Dos 的集成开发环境（IDE），可以完成编辑、编译、连接、调试、仿真等整个开发流程。开发人员可用 IDE 本身或其他编辑器编辑 C 语言或汇编源文件。然后分别由 C51 及 C51 编译器编译生成目标文件（.obj）。目标文件可由 LIB51 创建生成库文件，也可以与库文件一起经 L51 连接定位生成绝对目标文件(.abs)。abs 文件由 OH51 转换成标准的 hex 文件，以供调试器 dScope51 或 tScope51 使用进行源代码级调试，可由仿真器使用直接对目标板进行调试，也可以直接写入程序存储器如 EPROM 中。

Keil 的优点：Keil C51 生成目标代码效率非常之高，多数语句生成的汇编代码很紧凑，容易理解。在开发大型软件时更能体现高级语言的优势。

与汇编相比，C 语言在功能上、结构性、可读性、可维护性上有明显的优势，因而易学易用。用过汇编语言后再使用 C 语言来开发，体会更加深刻。

2. Keil MDK

Keil MDK 则是为基于 ARM 的微处理器设计的最广泛的软件开发工具，它包含了所有需要创建、编译以及调试嵌入式应用的组件，如图 1-3 所示。

Keil MDK 软件工具包括 MDK-Core 和 DS-MDK。前者是基于 μ Vision（只有 Windows 操作系统支持）的服务于 Cortex-M 系列设备的工具，后者则是基于 Eclipse（支持 Windows 和 Linux 操作系统）的服务于 Cortex-A 系列 32 位处理器（也向下包括 Cortex-M 系列）的工具。

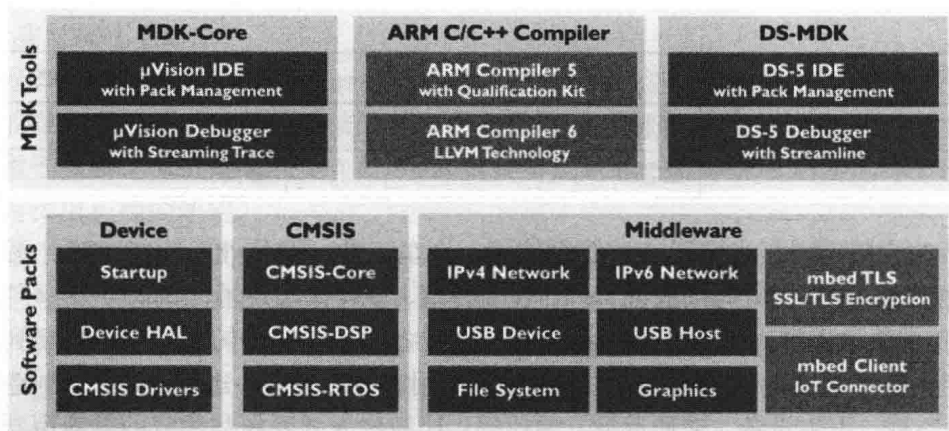


图 1-3 Keil MDK 产品组成图

1.3.5 物联网设备开发与 Mbed

1. 物联网设备开发

嵌入式微控制器以低成本、低功耗、易于开发等特点，被广泛应用于物联网设备当中。然而物联网设备作为近年来快速发展的新领域，又与传统的嵌入式系统应用场景，如家电、汽车、工业控制设备等相比，有其独特的关注点与开发方式。

(1) 数据连接。

物联网设备的最大特点是设备连接到网络。不论是既有设备的物联网化，还是新兴的智能硬件终端，都需要通过一定的硬件与对应的软件，连接到互联网上进行信息的交互，才能归类在物联网系统范围内。

一部分传统的嵌入式系统连接技术，如以太网、WiFi 等，借助网关设备实现互联网连接后，在物联网领域仍有很大的发挥空间。这些连接技术往往在硬件支持与软件协议方面十分成熟，中间设备丰富、成本较低，往往是既有设备进行物联网化的简便途径。另一些连接技术随着物联网的兴起，有针对性地进行升级以适应新的需求。蓝牙技术在经历了 4.0 版本的兼容性阵痛之后，由于其提供的低功耗特性，以及后续版本增加的短距离定位等新功能，已经成功地被基于网关的物联网系统所接受。以蜂窝网络为基础的 NB-IoT 技术，更新了对接入设备数量的更完善的支持，加上本身网络覆盖优异的特点，成为很多静止/低速、高密度物联网设备的优先选择。Thread 连接协议对 ZigBee 进行了新的标准化，改进了过去协议碎片化的问题，并借助 IPv6 实现海量设备的低功耗长连接，也具备了一定的发展潜力。与此同时，一些专门针对小型设备低速、低功耗设计的新技术也出现在这个领域中。例如基于 LoRa 的连接协议，在同等传输速度与功耗条件下，可以覆盖更远的传输距离，引起了业界的广泛关注。

不论是哪一种连接方式，软件协议栈都是程序开发当中比较复杂的一部分。程序需要通过一定的协议来建立连接，并对数据进行打包、加密以及解密、解包等处理。这些协议常利用连接技术提供方发布的协议栈库或者经过验证的开源协议栈库来实现，这些公用协议栈也会随着版本的更新不断完善功能并改进问题。

(2) 安全与加密。

随着物联网设备渗透到人们生活的方方面面，设备的安全也关系到了我们的人身安全与