

细说 Cortex-M0开发

冯迅 编著

以LPC1114为例

学习ARM Cortex-M0开发，从这里起航

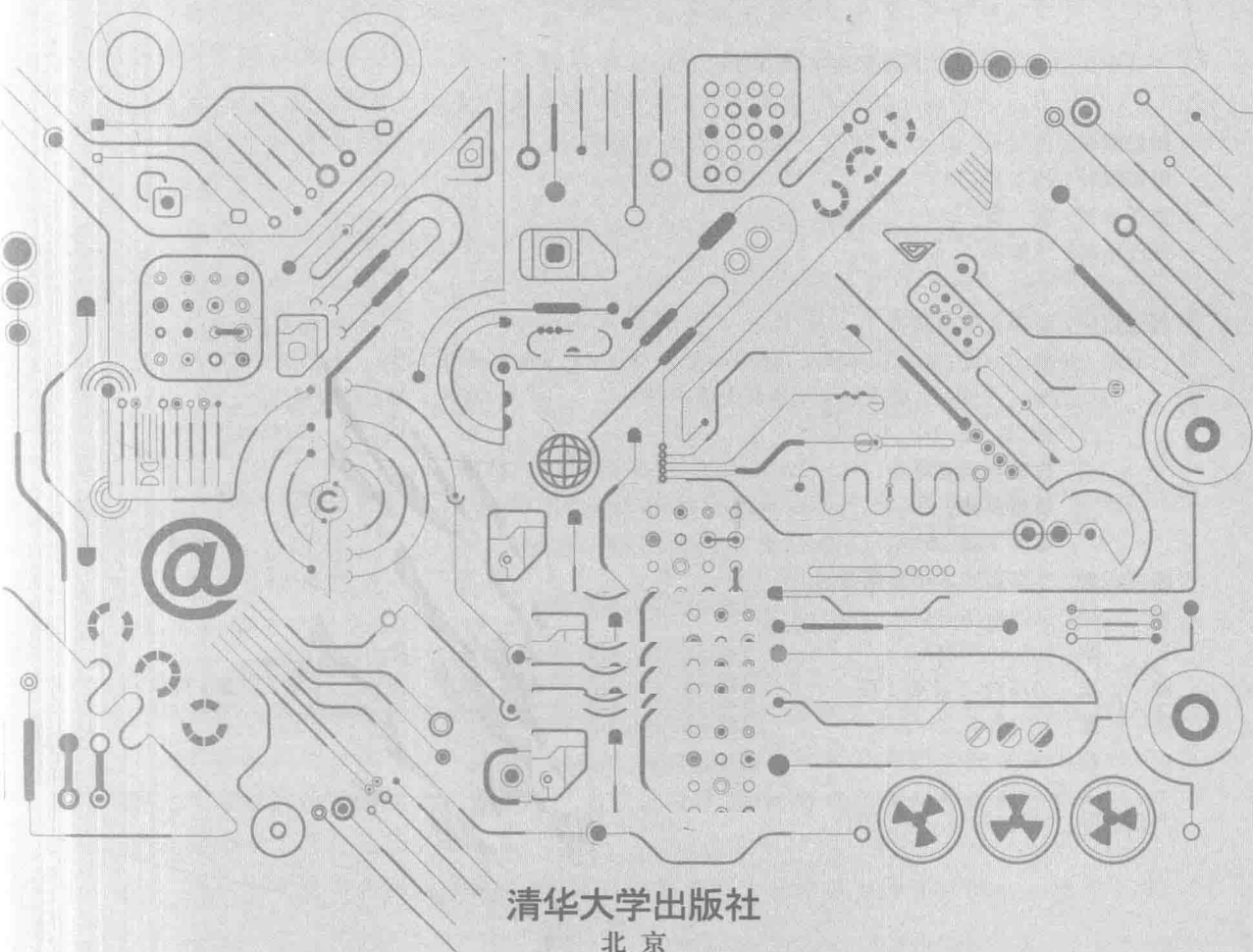
清华大学出版社



细说 Cortex-M0^{开发}

冯迅 编著

以LPC1114为例



清华大学出版社
北京

内 容 简 介

本书以恩智浦公司(NXP)的LPC1114芯片为例,详细讨论了ARM Cortex-M0处理器的开发过程。主要讲述了开发环境MDK-ARM的配置及LPC1114内部各个组成模块的结构及使用方法,包含GPIO端口、时钟源、NVIC中断系统、系统定时器、定时器/计数器、串口、A/D转换、I²C接口、SPI接口、看门狗、功耗管理及Flash编程固件等内容。全书以应用为主,针对LPC1114开发中可能遇到的问题都进行了详细的讨论。

书中所有代码无须任何修改就可直接使用,所有程序代码均在MDK-ARM4.22a环境下调试通过。

本书可作为高校及相关培训机构的教材,也可作为爱好者自学使用的书籍,还可作为嵌入式系统工程开发人员的参考手册。

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

细说Cortex-M0开发:以LPC1114为例/冯迅编著. —北京:清华大学出版社,2018
ISBN 978-7-302-48998-6

I. ①细… II. ①冯… III. ①微处理器—系统设计 IV. ①TP332

中国版本图书馆CIP数据核字(2017)第294406号

责任编辑:白立军

封面设计:杨玉兰

责任校对:梁毅

责任印制:王静怡

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

课件下载: <http://www.tup.com.cn>, 010-62795954

印 装 者:北京嘉实印刷有限公司

经 销:全国新华书店

开 本:185mm×260mm 印 张:20.25 字 数:481千字

版 次:2018年1月第1版 印 次:2018年1月第1次印刷

印 数:1~1500

定 价:59.00元

产品编号:070806-01

在当前嵌入式系统的低端应用领域,以 ARM Cortex-M0 为核心的芯片有着良好的市场前景。世界各大半导体厂商(如 NXP、Freescale、Atmel 等)都陆续推出了基于 Cortex-M0 内核的系列芯片。随着此类芯片的大量普及,其价格也越低,让广大开发者以 8 位芯片的价格获得了 32 位芯片的性能,大有取代传统低端芯片之势,所以,学习和使用 Cortex-M0 开发的人也越来越多,市场潜力较大。

2010 年,恩智浦(NXP)公司推出的基于 ARM Cortex-M0 内核的 LPC1100 系列芯片具有非常出色的性价比,随后又不断扩展升级,迅速引起业界的广泛关注。本书选取目前较为常用的 LPC1114 芯片,并把它当作一个 32 位的高级单片机来进行分析和讨论,本书在结构上也采用了传统单片机教材的形式。在编写过程中,均以官方提供的英文版数据手册为依据,书中所有引用的表格、图片、数据等都来自官方手册,但引用并非简单地对手册进行罗列翻译,而是在理解内容的基础上,结合初学者的认知规律和作者多年的教学实践经验,通过学习者易于接受的形式进行讲述,可提高学习兴趣,减小挫折感。书中讲述的内容符合初学者的认知规律,采取循序渐进的方式,力求解决在学习中可能遇到的问题,突出实际应用。

本书选取国内流行的 MDK-ARM 作为 LPC1114 的开发环境,用实例的方式讲述了开发中程序结构之间的关系,并详细剖析了头文件的用法,让读者不仅知其然还知其所以然。然后着重讲述 LPC1114 内部各组成模块的功能及实际应用,包含 GPIO 端口、时钟源、NVIC 中断系统、系统定时器、定时器/计数器、串口、A/D 转换、I²C 接口、SPI 接口、看门狗、功耗管理及 Flash 编程固件等内容。在每个章节的结尾,都会给出一个与该部分内容密切相关的实例,让学习者容易理解和掌握所讲授的内容,以此来提升学习效果。全书面向实际开发应用,注重对细节的分析,力求让读者能容易上手,快速掌握 Cortex-M0 的开发过程。

对于处理器的学习,从本质上来说就是学习处理器中各个寄存器的原理及其使用方法,为此本书讨论了 LPC1114 所用到的几乎所有寄存器,但并没有直接罗列所有的寄存器,而是把寄存器的学习放到了实际的例子当中,用到哪个就讨论哪个,这样能有效地避免单纯学习寄存器带来的枯燥。书中对寄存器的描述都先引用官方原文,然后再对其内容进行详细讨论。这样做不仅能提高资料的权威性,还可以避免翻译和理解上的歧义,有利于读者进行学习和参考。即便对于开发人员,本书也可作为资料手册进行查询,而不必再去翻阅官方数据手册。

书中所有程序代码及教学课件,都可以到清华大学出版社的官方网站上进行下载。

本书由云南师范大学信息学院冯迅编写,希望能对有志于学习 Cortex-M0 开发的读者起到积极的帮助作用。在此,对清华大学出版社提供的大力支持表示由衷的感谢!由于编者知识局限和时间仓促,书中不足之处在所难免,恳请广大读者批评指正!

冯 迅
2017年6月

第 1 章	LPC1114 及其开发环境简介	1
1.1	LPC1114 及其主要特性	1
1.2	MDK 开发环境及其配置	4
1.3	习题	9
第 2 章	时钟配置与仿真	10
2.1	LPC1114 的时钟及其配置	10
2.2	LPC1114 的时钟仿真	21
2.3	LPC1114 的时钟输出端口测试	25
2.4	习题	30
第 3 章	程序的编译与下载	31
3.1	编译程序	31
3.2	下载程序	32
3.2.1	Flash Magic	32
3.2.2	ULINK2	33
3.3	习题	36
第 4 章	通用输入输出端口及应用	37
4.1	实例引入	37
4.2	头文件解析	41
4.2.1	预定义分析	41
4.2.2	头文件应用	50
4.3	GPIO 应用详解	54
4.3.1	GPIO 端口操作分析	54
4.3.2	GPIO 端口寄存器	60
4.3.3	GPIO 引脚配置	65
4.4	GPIO 编程实践	72
4.5	习题	73

第 5 章 异常和中断系统及应用	74
5.1 异常	74
5.1.1 异常及向量地址	74
5.1.2 优先级	75
5.1.3 异常处理	75
5.2 中断系统	80
5.2.1 NVIC	80
5.2.2 寄存器及其映射	81
5.2.3 中断触发与处理	86
5.2.4 中断操作函数	86
5.3 外部中断	88
5.3.1 LPC1114 外部中断	88
5.3.2 外部中断入口函数	89
5.4 外部中断编程实践	90
5.5 习题	91
第 6 章 SysTick 定时器及应用	92
6.1 SysTick 定时器功能分析	92
6.1.1 内部结构	92
6.1.2 寄存器及其映射	93
6.1.3 初始值设置	95
6.2 SysTick 定时器编程实践	96
6.3 习题	97
第 7 章 通用定时器及应用	98
7.1 通用定时器概述	98
7.1.1 内部结构	98
7.1.2 寄存器及其映射	98
7.2 定时功能分析	101
7.2.1 寄存器配置	101
7.2.2 定时功能编程实践	106
7.3 计数功能分析	108
7.3.1 寄存器配置	108
7.3.2 计数功能编程实践	109
7.4 输入捕获功能分析	112
7.4.1 寄存器配置	113
7.4.2 输入捕获功能编程实践	114
7.5 PWM 功能分析	117

7.5.1	寄存器配置	117
7.5.2	PWM 功能编程实践	122
7.6	通用定时器综合实践	124
7.6.1	驱动 LCD1602 液晶屏	124
7.6.2	红外解码	129
7.7	习题	136
第 8 章	通用串行口及应用	137
8.1	UART 功能分析	137
8.1.1	内部结构	137
8.1.2	寄存器及其映射	137
8.1.3	寄存器配置	140
8.2	UART 综合实践	150
8.2.1	接收中断	150
8.2.2	发送端口状态	157
8.2.3	选择性启动	159
8.3	习题	164
第 9 章	A/D 转换及应用	165
9.1	A/D 转换功能分析	165
9.1.1	A/D 转换器的特性	165
9.1.2	寄存器及其映射	165
9.1.3	寄存器配置	167
9.1.4	注意事项	172
9.2	A/D 转换编程实践	173
9.3	习题	178
第 10 章	I²C 接口及应用	179
10.1	I ² C 接口功能分析	179
10.1.1	I ² C 总线	179
10.1.2	I ² C 接口的特性	180
10.1.3	寄存器及其映射	181
10.1.4	寄存器配置	183
10.2	I ² C 接口编程实践	190
10.3	习题	199
第 11 章	SPI 接口及应用	200
11.1	SPI 接口功能分析	200

11.1.1	SPI 接口特性	200
11.1.2	寄存器及其映射	200
11.1.3	寄存器配置	202
11.2	SPI 接口编程实践	210
11.3	习题	218
第 12 章	看门狗及应用	219
12.1	看门狗功能分析	219
12.1.1	内部结构及特性	219
12.1.2	寄存器及其映射	220
12.1.3	寄存器配置	221
12.1.4	注意事项	226
12.2	看门狗编程实践	227
12.3	习题	229
第 13 章	功耗管理及应用	230
13.1	功耗管理	230
13.1.1	PMU	230
13.1.2	SCR 寄存器	233
13.1.3	节能模式	235
13.2	功耗管理编程实践	242
13.3	习题	246
第 14 章	复位与 SWD 技术	247
14.1	复位	247
14.1.1	复位源	247
14.1.2	上电复位	249
14.1.3	外部引脚复位	250
14.1.4	掉电检测复位	251
14.2	SWD 调试	252
14.2.1	调试接口	252
14.2.2	连接方式	254
14.3	习题	254
第 15 章	Flash 编程固件	255
15.1	BootLoader	256
15.2	IAP	258
15.3	代码读保护	268

15.4	Flash 纠错	270
15.5	IAP 编程实践	270
15.6	习题	280
第 16 章	电子时钟实例	281
16.1	实例描述	281
16.2	电路原理图	281
16.3	实例分析	282
16.3.1	电路分析	282
16.3.2	功能分析	282
16.4	程序代码	282
16.5	代码说明	299
16.5.1	状态机编程	299
16.5.2	静态局部变量	302
16.5.3	其他说明	302
16.6	习题	302
附录 A	I²C 总线接口标准状态码	303
附录 B	标准 ASCII 码表	311
参考文献	313

LPC1114 及其开发环境简介

本章学习目标

- (1) 了解 LPC111x 系列微控制器的特性。
- (2) 掌握 MDK-ARM 开发环境及配置。

1.1 LPC1114 及其主要特性

LPC111x 系列微控制器是恩智浦(NXP)公司推出的基于 ARM Cortex-M0 内核的低功耗、32 位微控制器,具有高性能、低功耗、简单指令集、统一编址寻址等优点,相对于当前市场上的 8 位、16 位架构微控制器来说,它能有效地降低代码长度。

LPC111x 系列微控制器具有以下特点。

- (1) ARM Cortex-M0 处理器,工作频率高达 50MHz。
- (2) ARM Cortex-M0 内嵌向量中断控制器(NVIC)。
- (3) 32KB(LPC1114)、4KB(LPC1113)、16KB(LPC1112)或 8KB(LPC1111)片上 Flash 可编程存储器。
- (4) 高达 8KB 的 SRAM。
- (5) 通过片上引导(BootLoader)软件实现现场编程(ISP)和在线编程(IAP)。
- (6) 串行接口。
 - ① 有分数波特率发生器,内部 FIFO,支持 RS-485/EIA-485 总线和 Modem 控制的 UART。
 - ② 最多可有两个具有 SSP 特性的、带 FIFO 的 SPI 控制器,具有很好的多协议兼容性(第二个 SPI 只在 LQFP48 和 PLCC44 封装中存在)。
 - ③ I²C 总线接口支持全速 I²C 总线规格和增强快速模式(速率可达到 1Mb/s,具有多地址识别监听模式)。
- (7) 其他外设。
 - ① 多达 42 个带有可配置上拉/下拉电阻的 GPIO 引脚。
 - ② 每个引脚有大电流(20mA)驱动输出。
 - ③ 两个 I²C 总线引脚在增强快速模式时,为大电流灌入驱动(20mA)。
 - ④ 4 个通用定时器(共 4 个捕获输入和 13 个比较输出)。
 - ⑤ 看门狗定时器(WDT)。
 - ⑥ 系统滴答定时器。

(8) 串行线调试(SWD)。

本书以 LPC1114 作为 ARM Cortex-M0 的学习选型。LPC1114 是恩智浦公司推出的基于 ARM Cortex-M0 内核的一个 32 位低功耗微控制器,其内部拥有 32KB 的 Flash ROM、8KB 的 RAM,低电压供电(1.8~3.6V),速度快(0.9DMIPS/MHz),功能强大,价格低廉,应用非常广泛,其 LQFP48 封装的实物外形如图 1-1 所示。

LPC1114 芯片一共有 48 根引脚,其中 42 根是通用输入输出引脚(GPIO),分为 PIO0 口(PIO0_0~PIO0_11)12 根脚,PIO1 口(PIO1_0~PIO1_11)12 根脚,PIO2 口(PIO2_0~PIO2_11)12 根脚,PIO3 口(PIO3_0~PIO3_5)6 根脚。余下的 6 根引脚分别是两个晶振引脚、两个正电源引脚(系统、A/D)和两个地引脚(系统、A/D)。在要求不严格的情况下,系统电源和 A/D 电源可以接在一起。图 1-2 给出了一个 LQFP48 封装的 LPC1114 微控制器引脚分布情况。

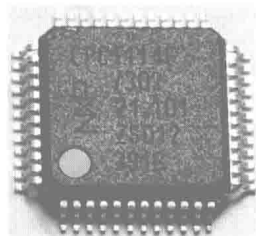


图 1-1 LQFP48 封装的 LPC1114

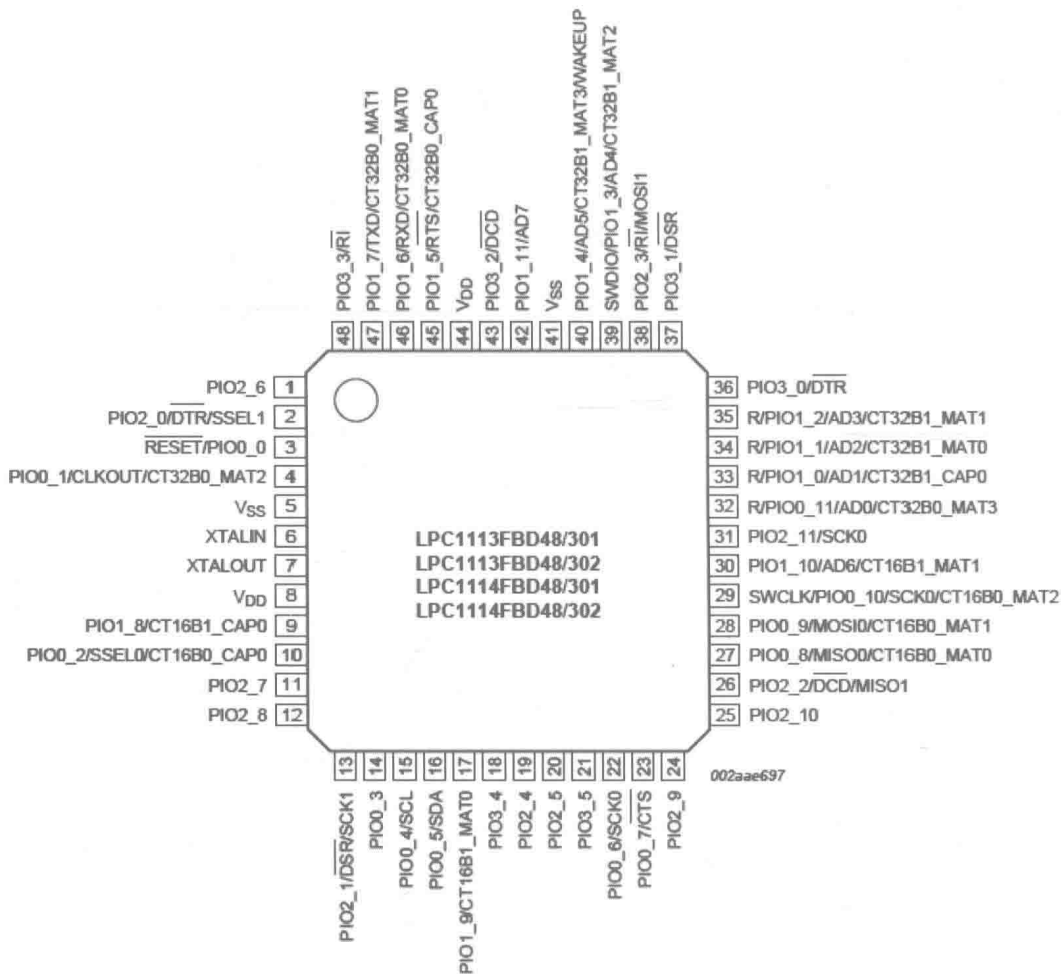


图 1-2 LPC1114 引脚图(LQFP48)

从图 1-2 中可以看到, LPC1114 的大多数引脚都具有复用功能, 甚至芯片的复位端 (RESET) 都与引脚 PIO0_0 复用在一起了 (第 3 脚), 所以 LPC1114 芯片没有独立的复位引脚, 不过在默认状态下, 第 3 脚是复位端口 (即第一功能为复位功能)。

LPC1114 系列内部自带有 12MHz 的 RC 振荡器, 并且在其上电 (或复位) 时默认使用片内的 RC 为主时钟源。所以 LPC1114 在默认情况下, 上电就可运行。不过内部 RC 振荡器的精度并不高 (一般为 1%), 所以在要求较高时还是要使用外部晶振来作为振荡时钟。图 1-3 给出了一个使用外部晶振的最小系统。

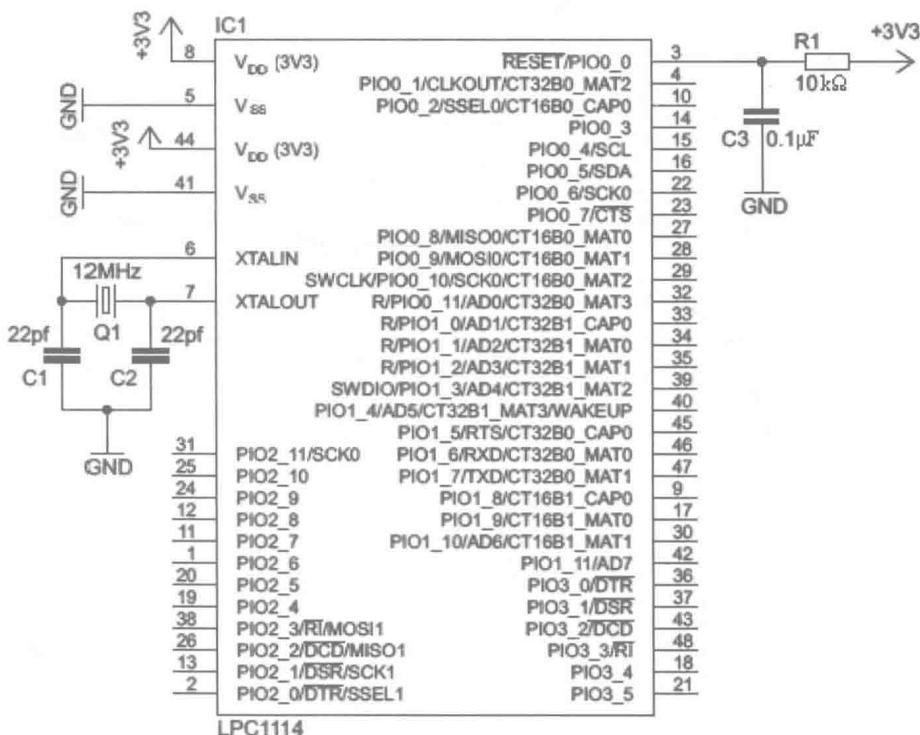


图 1-3 LPC1114 最小系统原理图

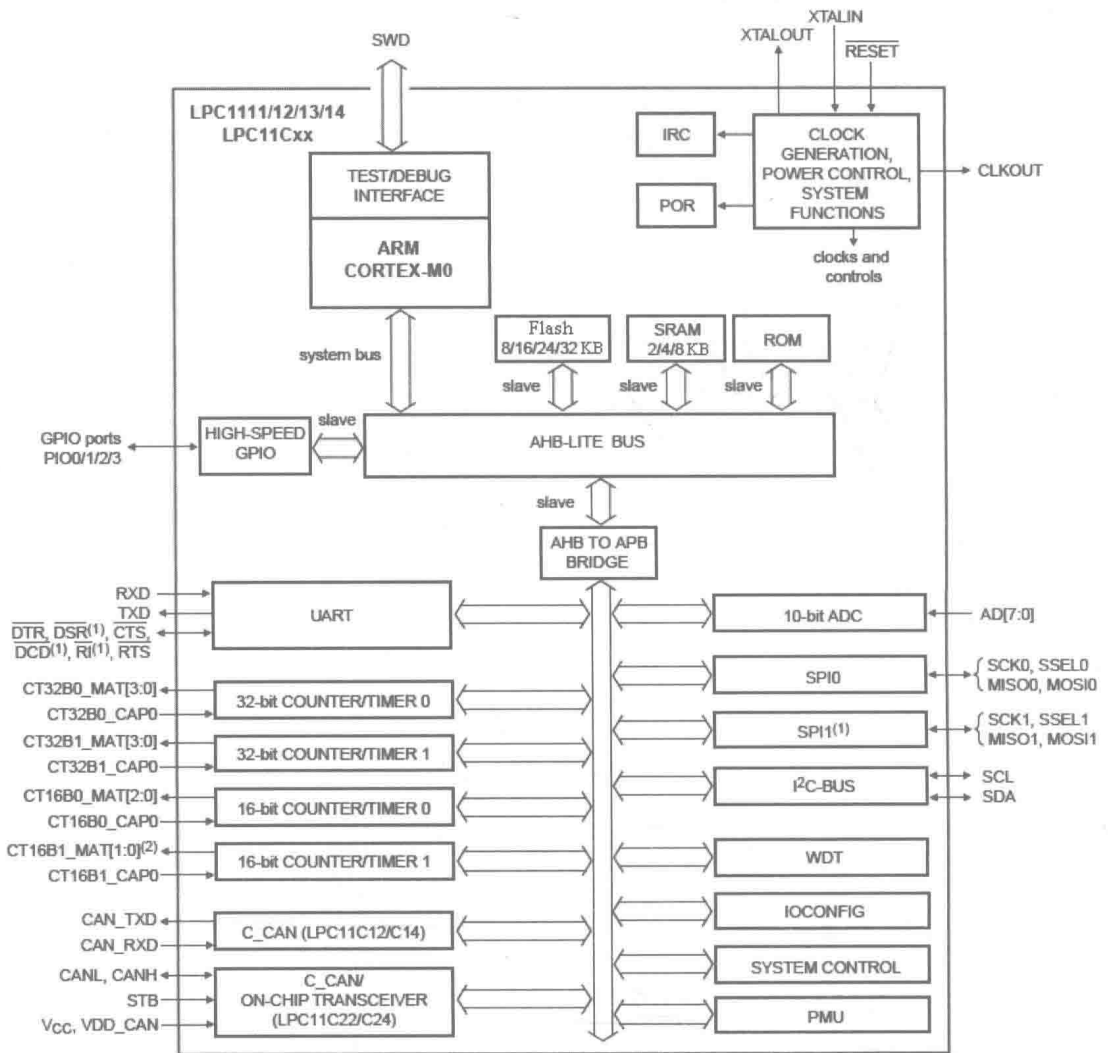
图 1-4 给出了 LPC1114 芯片的内部结构。

从图 1-4 中可以看到, 在 LPC1114 微控制器芯片中, 最核心的部分是一个 Cortex-M0 的 ARM 内核。

Cortex-M0 内核是 ARM 公司为适应低端市场的需求而专门设计的一个入门级的 32 位 ARM 处理器, 它目前被广泛地应用于各类嵌入式系统中。该处理器具有结构简单、容易学习和编程、功耗极低、运算效率高、代码密度出色、高性能中断处理、向上兼容 Cortex-M 系列处理器等特性。

Cortex-M0 采用 ARMv6-M 结构, 基于 16 位 Thumb 指令集, 并包含 Thumb-2 技术。所以能提供一个现代 32 位体系结构处理器所希望的出色性能, 代码密度比其他 8 位和 16 位微控制器都要高。

ARM Cortex-M0 处理器基于一个高集成度、低功耗的 32 位处理器内核, 采用 3 级流水线的冯·诺依曼结构。通过简单、功能强大的指令集以及全面优化的设计 (提供包括一



- (1) LQFP48/PLCC44 packages only.
- (2) Not available on LPC11/C22/C24.

图 1-4 LPC1114 内部结构图

个单周期乘法器在内的高端处理硬件), ARM Cortex-M0 处理器可实现极高的能效。

NXP 公司基于 ARM Cortex-M0 内核设计了 LPC11xx 系列微控制器, 并加入了大量功能丰富的外设和通信接口, 给嵌入式开发者提供了极大的便利。

1.2 MDK 开发环境及其配置

开发基于 LPC1114 的系统要选择一个合适的开发工具。对于 NXP 系列 CPU, 恩智浦公司专门委托 code_red 公司为其量身定制了一款基于开源项目 Eclipse 的开发工具, 称为 LPCXpresso。该软件可以通过 code_red 公司网站注册后免费下载并激活。其他的开发工具, 基本上都是基于 ARM 的通用开发环境, 常见的有 ARM 公司著名的 Real

View MDK(MDK-ARM)、IAR 公司的 EWARM 等。本书选用国内使用非常广泛的 MDK-ARM 作为 LPC1114 的开发工具,其他开发工具读者可自行研究使用,它们都有各自的特点。

Real View MDK 的前身其实就是著名的 Keil μ Vision,后来它被 ARM 公司收购后就结合 ARM 系统推出了 MDK-ARM。熟悉 Keil μ Vision 的用户会发现两者的界面基本上是一样的,使用起来非常方便。

本书使用的是 MDK-ARM4.22a 版本,它结合了程序编辑、编译、查错、调试、仿真等功能为一体,功能强大,使用方便,熟悉 51 单片机开发的读者可以很快上手。图 1-5 是它的运行界面,可看出它基本上就是 Keil C51 的界面。

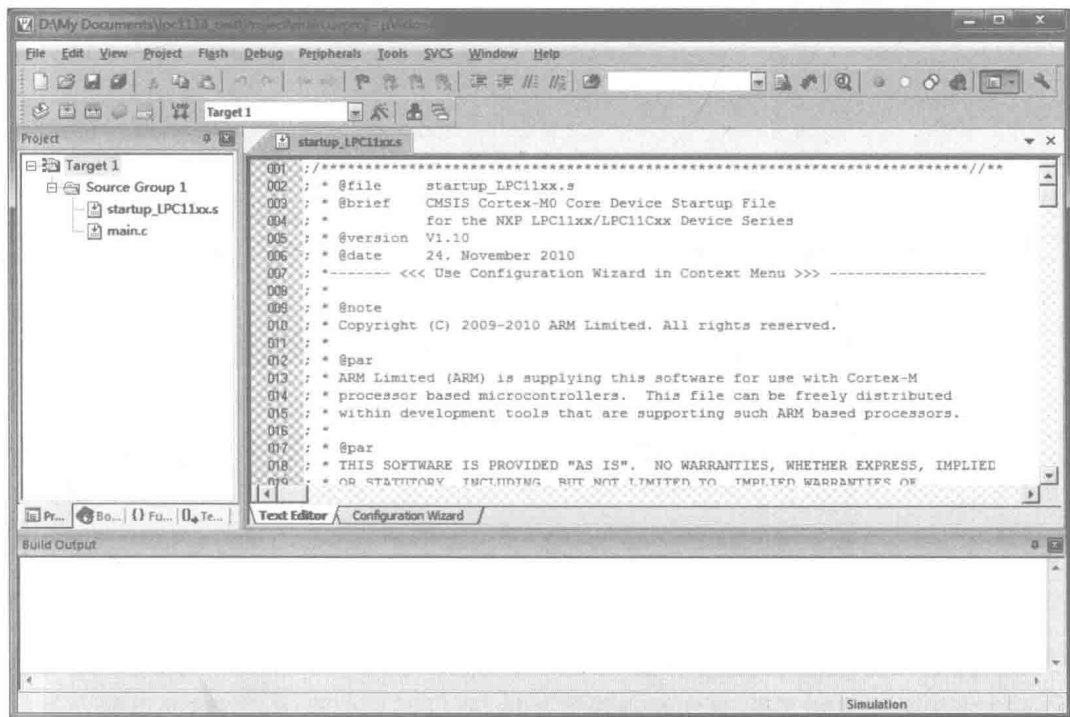


图 1-5 MDK-ARM 的运行界面

MDK-ARM 开发环境的配置非常简单,下面就用它来具体配置一个 LPC1114 的开发环境。

配置步骤如下。

- (1) 安装 MDK-ARM4.22a 版本,输入许可证码将其激活(详略)。
- (2) 新建一个工程,单击 Project→New μ Vision Project,在弹出的对话框中选择一个路径并输入一个工程名称(如 test,默认扩展名为 μ vproj),单击“保存”按钮。
- (3) 在弹出的器件选择对话框中选择 NXP 下的 LPC1114/301,并单击 OK 按钮(见图 1-6)。
- (4) 接下来会弹出一个如图 1-7 所示的对话框,询问是否要把启动代码(Startup Code)加入到工程中,单击“是”按钮。

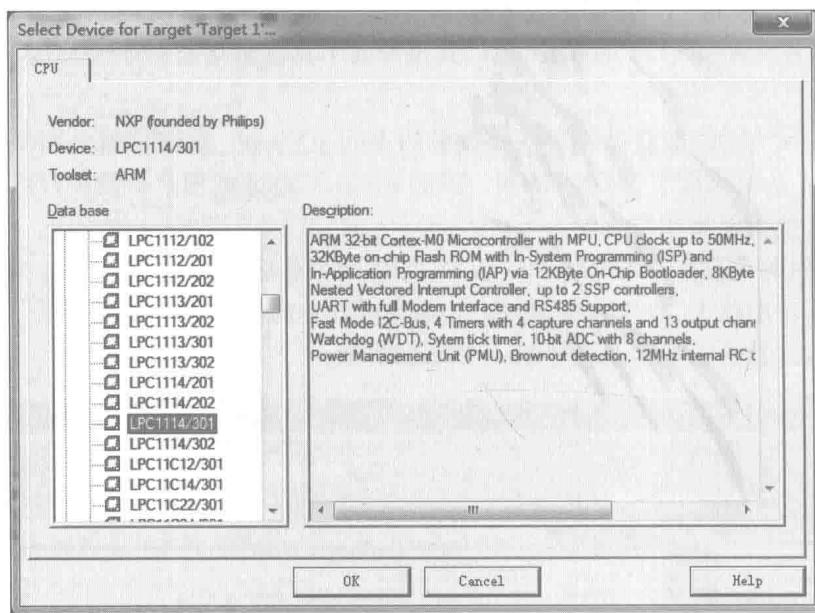


图 1-6 器件选择对话框

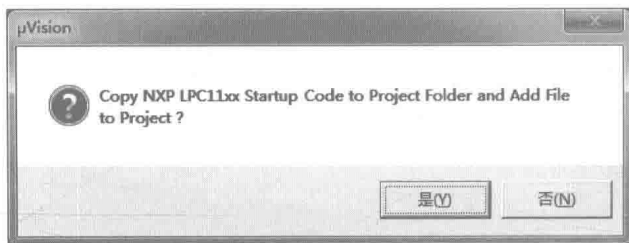


图 1-7 复制启动代码对话框

(5) 完成后就可进入到工作界面,如图 1-8 所示。

(6) 由于是新建的工程,其中没有任何文件,所以下一步要加入一些必要的文件(如头文件等),同时要新建主程序文件并把它也添加到工程中来。

(7) 新建主程序文件,单击 File→New,可看到在工作区域内新建了一个默认名称为 Text1 的文本文件,如图 1-9 所示。

(8) 接下来保存该主程序文件,单击 File→Save,在弹出的对话框中选择好保存路径,然后在文件名中为主程序文件取一个名称,注意名称一定要加上扩展名,如本例中为 c,单击“保存”按钮。

(9) 最后一步要把刚才新建的主程序文件添加到工程中来,单击左侧 Target1 中的加号把它展开,然后在其下面的 Source Group 1 上右击,在弹出的快捷菜单中选择 Add Files to Group 'Source Group 1'一项,如图 1-10 所示。

(10) 在弹出的对话框中找到刚才保存的主程序文件,单击 Add 按钮将其添加到工程中(见图 1-11)。这里要注意,若找不到主程序文件可能有以下几个原因:一是刚才保存时没有为主程序文件加上扩展名 c;二是对话框中的筛选条件不对(文件类型应为 C

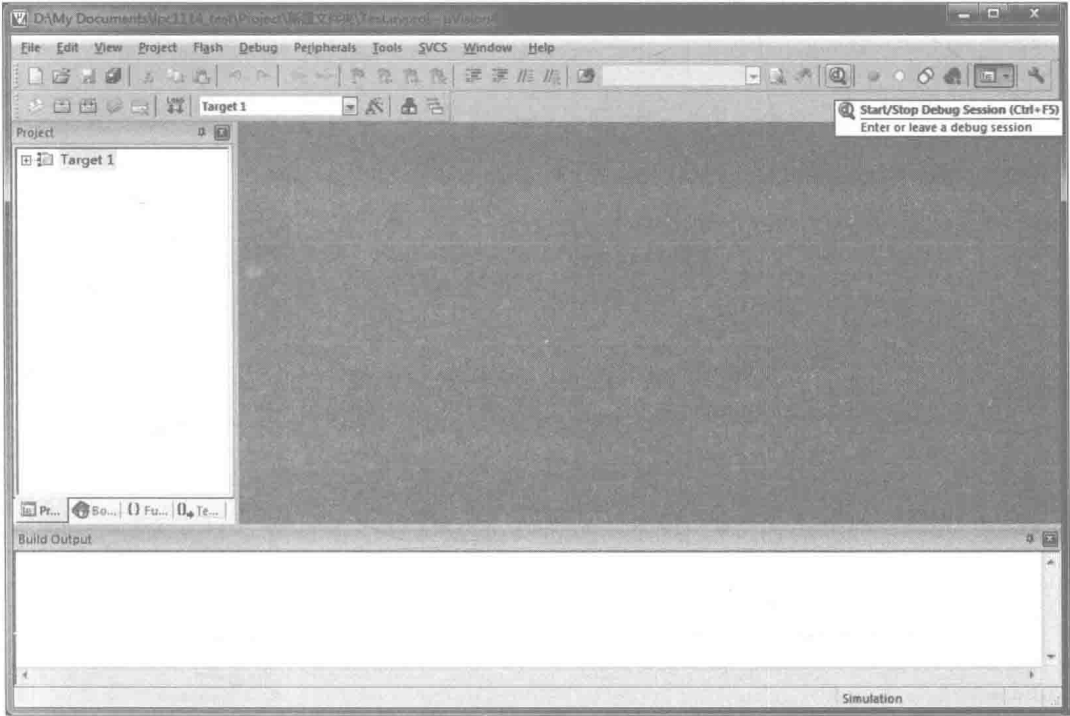


图 1-8 工作界面

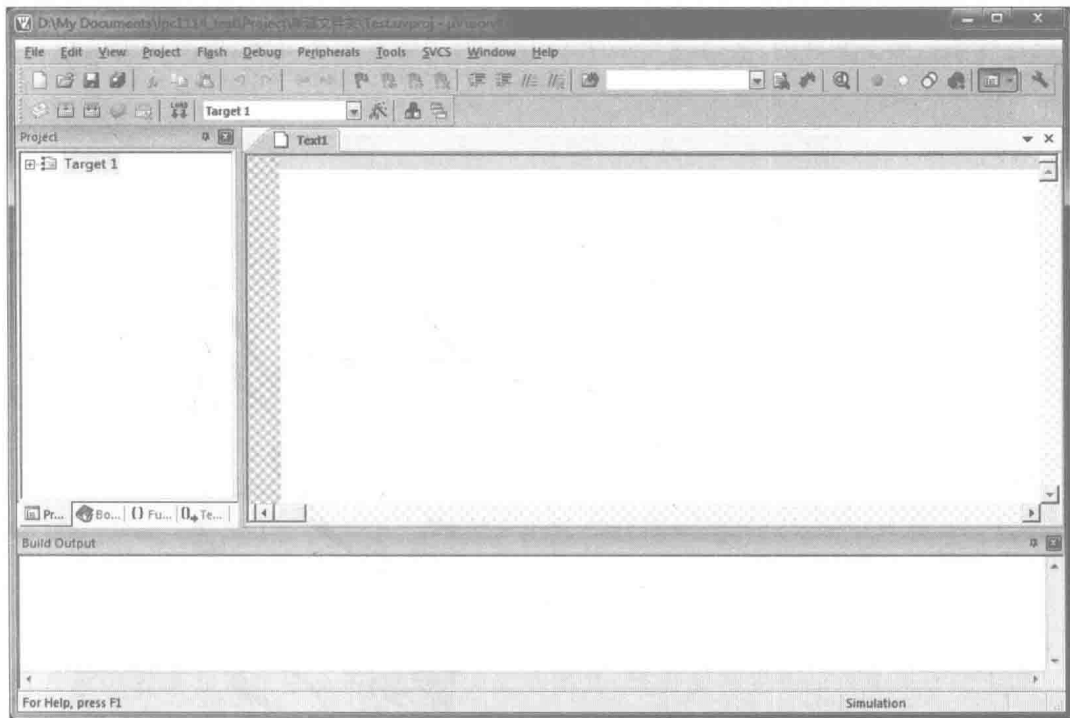


图 1-9 新建主程序文本