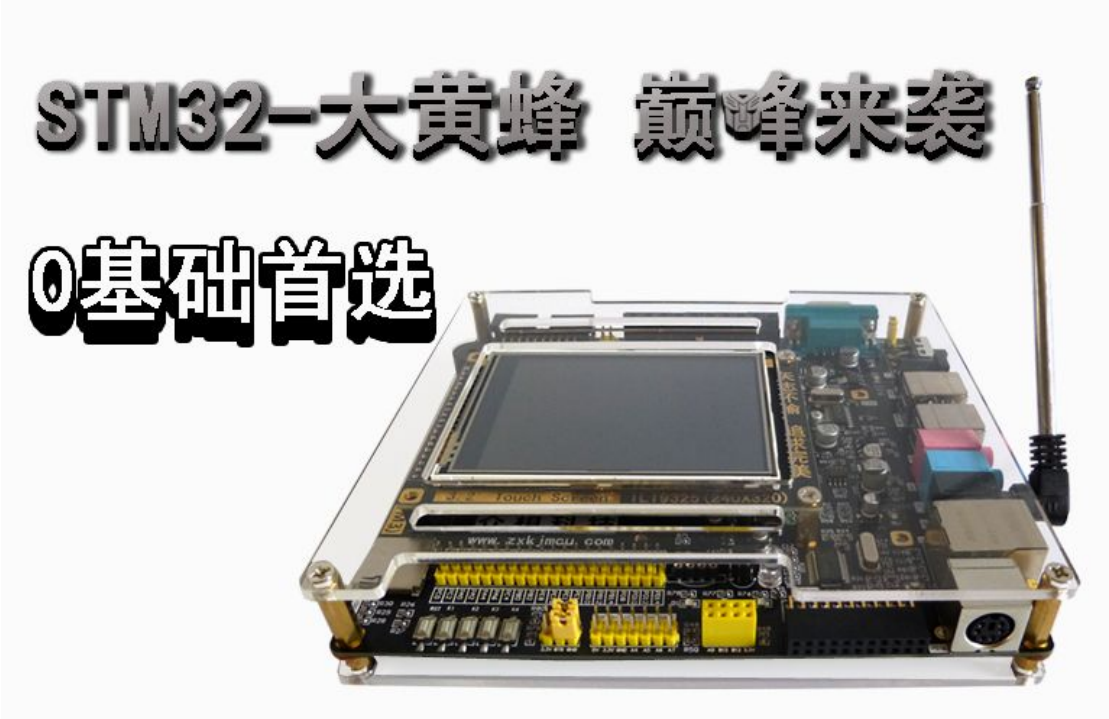
学 ARM 从 STM32 开始

STM32 开发板库函数教程--前言



官方网站：http://www.zxkjmcu.com
官方店铺：http://zxkjmcu.taobao.com
官方论坛：http://bbs.zxkjmcu.com
刘洋课堂：http://school.zxkjmcu.com

前 言

1.学习 STM32 需要哪些基础

首先要对 C 语言有一定的了解，不用学到精通，只要会使用一些简单的命令语句。比如赋值语句、if 语句、for 循环语句、whil 语句、一维数组、二维数组等。还有一点需要重点学习的是函数的定义、调用，参数的传递与参数的返回。推荐给大家一本学 C 语言的书籍<<谭浩强 c 语言程序设计>>,这本书也是很多大学里的教材，写的很详细，我上大学时也是学的这本书。

当你学会了 C 语言，说明你已经掌握了软件的基础了，接下来需要学习一些硬件的基础知识。可以到书店买两本数字电路与模拟电路相关的书籍，建议你不用学的太深，只要简单的了解能看懂原理图就可以了，等到你真正到工作岗位上，大的公司里软件开发人员和硬件设计人员都是分开的，只有专心做一件事情才能做的精益求精，有一个好的发展。

如果你以前学过 51 单片机，你一定熟悉 C 语言、模拟电路和数字电路。那么你就可以直接学习 STM32。需要注意的是我们以前学习 51 单片机时都是直接操作寄存器，而现在我们学习 STM32 大部分都是调用库函数。那么寄存器和库函数有什么不同呢？请大家看后面我从网上摘取的一段文字。

如果你以前没学过 51 单片机或者没接触过单片机，那么你可以到我们的网站上看看我们以前讲过的一套<51、AVR、PIC 三合一视频教程>这套视频教程里详细的讲解了这三种单片机的使用方法，作为学习 STM32 的基础已经足够用了。

2.STM32 的特点

Cortex-M3 处理器是一个低功耗的处理器，具有门数少，中断延迟小，调试容易等特点。它是为功耗和价格敏感的应用领域而专门设计的、具有较高性能的处理器，应用范围可从低端微控制器到复杂 SoC。

Cortex-M3 处理器使用了 ARM v7-M 体系结构，是一个可综合的、高度可配置的处理器。它包含了一个高效的哈佛结构三级流水线，可提供 1.25DMIPS/MHz 的性能。在一个具有 32 个物理中断的标准处理器上实现(0.13umMetro@50MHz),达到了突出的 0.06mW/MHz 能效比。

为了降低成本，Cortex-M3 处理器采用了与系统部件紧耦合的实现方法，来缩小芯片面积，其内核面积比现有的三级流水线内核缩小了 30%。Cortex-M3 处理器实现了 Thumb-2 指令集架构，具有很高的代码密度，可降低存储器需求，并能达到非常接近 32 位 ARM 指令集的性能。

对于系统和软件开发，Cortex-M3 处理器具有以下优势：

1.小的处理器内核、系统和存储器，可降低器件成本；

2.完整的电源管理，很低的功耗；

3.突出的处理器性能，可满足挑战性的应用需求；

4.快速的中断处理，满足高速、临界的控制应用；

5.可选的存储器保护单元(MPU)，提供平台级的安全性；

6.增强的系统调试功能，可以加快开发进程；

7.没有汇编代码要求，简化系统开发；

8.宽广的适用范围：从超低成本微控制器到高性能 Soc。

Cortex-M3 处理器在高性能内核基础上，集成了多种系统外设，可以满足不同应用对成本和性能的要求。处理器是全部可综合、高度可定制的(包括物理中断、系统调试等)，Cortex-M3 还有一个可选的细粒度的(fine-granularity)存储器保护单元(MPU)和一个嵌入式跟踪宏单元(ETM).

3.如何学习 STM32

如果你去问业内人士，怎样学习 STM32,他们都会说多实践，做多了自然就会了。可是我觉得对于一名初学者，没有一点的理论基础，根本就无从下手，不知道从何做起。我们常说实践是在有理论的指导下进行的，没有理论的实践是蛮干，没有实践的理论那是空谈。那么怎样获得理论知识那？最好的办法就是看视频教程与配套的书籍。建议初学者多看视频教程，一遍不会就看两遍。然后再仔细看看配套的书籍，当我们有了理论基础，就可以动手实践了。实践也要由小到大，由浅入深的进行，别总想一步登天。知识都是一点一滴的逐渐积累起来的。

我们的视频教程是站着初学者的角度来讲解的，整套视频教程都是以边讲边写的风格由浅入深的进行。现在已经在业内得到了广泛的好评，有兴趣的同学可以先到我们的网站上下载，下载地址是：www.zxkjmcu.com，网站上还有很多的资料，可供大家参考学习使用。整套视频教程主要分为三篇：

1.基础篇，可以免费观看，主要讲解的是 STM32F103VET6 芯片内部的资源。包括(TIMER、CAN、ADC、DAC、RTC、DMA、USART 等)

2.外设篇，需要购买开发板才能观看。主要讲解的是外围芯片的使用方法。包括(24C04、FLASH 芯片、红外发射与接收、MP3、FM 收音机、USB 通讯、以太网、触摸屏等)。

3.系统篇，也是需要购买开发板才能观看。主要讲解 ucos II 实时操作系统和 ucgui 图形支持系统的移植与使用方法。

只要同学能认真的把我们这套视频教程与配套的书籍都学完，那么你一定会成为一名 STM32 高手。

4.学完 STM32 能做什么

1.销售终端

银行的读卡机、收银机、热敏打印机、票据验证、包裹跟踪、自动售货机。

2.身份识别设备

安全和生物特征识别、公路自动收费系统。

3.工业自动化

现场数据采集器、电表、可编程逻辑控制器(PLC)、工业缝纫机。

4. 消费电子

计算机外设、游戏手柄、玩具、万能遥控器、卫星收音机、

5. 建筑安防/消防/HVAC

报警系统、控制面板。

6. 医疗

心脏监控、便携试仪器。

7. 通信领域

同声翻译系统、光纤接入控制、3G 基站监控。

8 家电

电动自行车、变频空调、洗衣机。

9. 仪器仪表

电子秤、电表、水表。

寄存器与库函数的区别

首先，两个都是 C 语言。从 51 过渡过来的，就先说寄存器操作。每个 MCU 都有自己的寄存器，51 是功能比较简单的一种，相应的寄存器也比较少，我们常用的就那么几个，像 P0、P1、SMOD、TMOD 之类的，这些存在于标准头文件 reg.h 里面。因为少，所以大家就直接去操作了。每一位对应的意义随便翻一下手册就看得到，甚至做几个小项目就记的很清楚了。所以做 51 开发的时候大多数都是直接操作寄存器。

到了 STM32，原理一样，也是有自己的寄存器，但是作为一款 ARM 内核的芯片，功能多了很多，寄存器自然也就多了很多，STM32 的手册有 1000 多页，这时候想去像 51 那样记住每个寄存器已经不现实了，所以 ST 的工程师就给大家提供了库函数这么一个东西。库函数里面把 STM32 的所有寄存器用结构体一一对应并且封装起来，而且提供了基本的配置函数。我们要去操作配置某个外设的时候不需要再去翻眼花缭乱的数据手册，直接找到库函数描述拿来就可以用，这样就能把精力放在逻辑代码的开发上，而不是去费力的研究一个芯片的外设要怎么配置寄存器才能驱动起来。

简单讲就是这些了，库函数是为了让开发者从大量繁琐的寄存器操作中脱离出来的一个文件包，在使用一个外设的时候让开发者直接

去调用相应的驱动函数而不是自己去翻手册一个一个配置寄存器。

有人说用库函数掌握不到芯片的精髓，仁者见仁智者见智。熟悉一款芯片是在不断的开发使用中逐渐了解并掌握的，调试的过程中会遇到很多问题，会要求我们去跟踪相关寄存器的状态，在整个框架都已经建立起来的基础上再去对照手册做具体到寄存器每一位的分析，代码对照现象，很快就能积累起来经验。