**1 Tiva LaunchPad简介**

TM4C**123**GH6PM是TI公司推出的一款32位基于ARM Cortex-M4的处理器，主频80MHz，256kB Flash，32kB SRAM，具有USB Host，Device和OTG的能力。

TivaC LaunchPad是基于TM4C123GH6PM控制器的实验板卡，自带仿真器，连接上USB即可进行Corte-M4的学习。

板子上带2个用户按键和1个三色的LED灯；对外引出的IO口符合BoosterPack 40Pin标准定义（详见TI的文档），也就是说可以兼容MSP430 LaunchPad板卡，整个板卡示意图如下：



拿到TivaC LAUNCHPAD板子时，不用急于上电操作，“磨刀不误砍柴工”，先花点时间大致了解下板卡的硬件电路连接。使用前，只要了解大概即可。

**1.1 电源**

板子通过USB口供电。有两个USB口（Device和Debug；USB座子边上有丝印字符表示）， 这两个USB口都可以给板子供电，我们选择接Debug。 板子上有一个绿色的LED灯，用来指示3.3V供电。

对外输出两路电压：

3.3VDC 最大300mA；

5.0VDC 输出能力23mA~323mA，与3.3VDC的使用有关。3.3V用的多了，供这边使用的就少了。

**1.2 复位电路**

低电平有效复位。当按键按下时，为低电平。



**1.3 时钟电路**

板卡上有两个晶振。16MHz晶振（Y2）提供给处理器，通过内部PLL，倍频后再分频给内核和外设使用； 晶振32.768KHz（Y1）用于休眠系统的时钟源。



**1.4 用户按键**

按键按下时，IO口接地，为低电平；

**GPIO-PF4---->按键SW1**

**GPIO-PF0---->按键SW2**





**1.5 三色LED灯显示**

LaunchPad板子上带一个三色的LED灯，位于复位按键下边。

IO口输出高电平时，三极管导通，LED灯亮。

GPIO-PF1---->RGB LED (红色)

GPIO-PF2---->RGB LED (蓝色)

GPIO-PF3---->RGB LED (绿色)



**1.6 虚拟串口**

当LaunchPad板卡连接到电脑时，作为调试器的同时，也会有一个虚拟的串口可以使用。处理器上使用的串口引脚为：

 PA0 ---->U0RX

 PA1 ---->U0TX



打开设备管理器，记下虚拟串口的端口号，例如下图中为COM37。同时，请确保驱动已经安装成功；



**1.7 调试接口**

LaunchPad板自带一个板上仿真接口ICDI（In-Circuit Debug Interface），通过USB连接电脑后即可使用。ICDI可用于编程或者调试TM4C123GH6PM。支持LM Flash Programmer或者兼容的工具。

其它的调试接口可以连接到SWD（Serial Wire Debug）和SWO。注意，ICDI仅支持JTAG调试。

 PC0---->TCK/SWCLK

 PC1---->TMS/SWDIO

 PC2---->TDI

 PC3---->TDO/SWO

**1.8 USB接口电路**

一般情况，可以直接连接USB接口电路。

要使能OTG功能，需要焊接板子上的电阻R25和R29（两个均为0欧）。电阻将USB ID、USB Vbus信号分别连接到芯片的PB0、PB1上；同时，引脚PB0和PB1必须配置成USB引脚模式。因为PB0、PB1也作为扩展引脚上，为了避免器件损坏，当R25和R29电阻焊接时，外接BoosterPack要特别注意PB0和PB1这两跟引脚不能使用。

**2 Tiva的开发环境搭建**

在前一篇中，我们已经了解了TM4C123GH6PM LaunchPad板卡的基本硬件电路。TI推出的LaunchPad系列板卡，可以非常方便的进行处理器的学习与开发。在硬件上，有了LaunchPad板卡和电脑，外加一根USB线连接，就可以开始嵌入式处理器的编程学习了。当然，需要先在电脑上安装一些开发软件。



经过前边的几个步骤，我们已经把Tiva处理器的开发环境搭建好了。

安装驱动把Tiva LaunchPad连接到电脑，上电。 选择驱动程序进行安装。

在学习编程时，主要分为以下3个大的阶段进行：

**1)** 直接下载已经编译好的**BIN**文件

我们要做的就是把Bin文件下载到芯片上就OK了，做的是代码烧录的工作。熟悉下LM Flash Programmer工具的使用。

**2)** 编译现有的代码工程

将现有的工程范例代码，导入到Keil中，编译下载看现象，并能在工程上做一些简单的修改操作。

**3)** 新建空白工程，编译下载

通过新建工程、配置工程、编写代码等操作，一步步打造符合自己要求的工程。

**3.TI ARM CortexMCU的开发方式**

这里所说的TI ARM Cortex MCU的开发方式是指下载并运行在MCU上的软件的开发方式。开发方式的含义之一是开发语言的选择。MCU上的软件通常有两类开发语言，一类是汇编指令，另一类是高级语言（通常是C或者C++）。汇编指令会因MCU种类不同而不同，程序逻辑控制困难，代码可读性及可维护性都不佳，而现在编译器的效率越来越高，所以采用汇编指令作为开发工具的机会越来越少，只有少数程序或者程序的某个模块对代码的效率要求极高时才会采用汇编指令作为程序开发语言。C语言已经成为当前大部分MCU软件开发的主要语言。随着C++的编译器的进步和MCU性能的提高，在项目复杂度和MCU性能都较高的应用越来越多地利用面向对象的C++语言作为开发工具，以获得更优的项目管理特性。

开发方式的另一含义是指利用TI提供的哪些库（**Library**），或者利用库的哪些部分。所谓的库，它是一组功能函数的集合，这个集合里函数的功能都已实现，并有一个易于调用的接口1。库为软件开发人员方便使用其它开发人员已完成的功能（代码）提供了一种载体，使代码具有可重用性。也就是说你也可以将你实现的一系列代码做成库的形式，来供其它人员使用，当然也包括你自己将来使用这个库，称之谓可重用性。其实TivaWare就是由多个库组合成的库的集合。

TivaWare是一个不断扩展的库，TI公司、第三方以及开源社区都为此做出贡献。TivaWare为用户使用它提供了大量的说明文档和示例代码。

**3.1 BootLoader简介**

(1) BootLoader的概念

BootLoader，顾名思义，是Boot（启动）的时候起到Loader（加载器）作用的代码，即系统刚一上电启动，最先被执行的一段代码，根据这段代码的安排，系统的其余部分会被它加载（Load）并运行。BootLoader被放在系统的某一个特定的地方，每次系统一上电，它都会比任何其它程序先运行。

Tiva的BootLoader

在PC里，BIOS通常存放在主板上的一块ROM里，PC一上电默认找到这块ROM内的程序执行；在Tiva系列MCU系统里，起到类似PC里ROM的功能的是闪存（Flash存储器，简称Flash）, BootLoader是存放在Flash起始地址处的一小段代码，占据默认大小为2K字节的空间。Tiva的BootLoader有两种可配置的加载功能，其一是加载用户应用程序（Application）；其次是与其它设备建立标准通信通路，并从该设备上加载新的系统软件来更新（Update）原有的系统。更新的过程通过对固件（Firmware）的烧写来实现。烧写固件也是那些经常更新手机系统的用户比较熟悉的术语。系统上电后BootLoader内的启动代码（Start-up Code）将先被执行，进行一系列的初始化操作后，根据预先设定的条件，选择执行用户应用程序或更新控制程序（Updater）。Cortex M4F系列单片机硬件启动原理如下：硬件复位时，NVIC\_VTABLE复位为0，向量表默认位于Flash空间起始地址处（0x00000000）。

(2) BootLoader的启动代码（Startup Code）

BootLoader的启动代码是一小段精简代码，它要完成（1）配置向量表；（2）初始化存储空间的分布；（3）复制BootLoader代码到SRAM；（4）从SRAM中执行代码等一系列操作。

由于配置向量表和初始化存储空间的分布与用户使用的开发工具的编译器和链接器有关，所以每种工具都有它自己对应的启动代码实现和链接配置文档。

TivaWare定义了一组对寄存器设置的宏，这些宏的定义规则如下：

这些宏的定义也在以MCU型号命名的头文件中（本实验套件对应着tm4c123gh6pm.h文件）。在*TivaWare*安装目录中的*inc*文件夹下，有大量以*“hw\_\*.h”*命名的头文件，它们是用来使用软件驱动模式编程时要包含的头文件。

“tm4c123gh6pm.h”，它有我们要用到的宏定义；2）“stdint.h”, 它有int数据类型的重定义，如果你的程序中还使用了别的数据类型，比如bool型，那你还要包含“stdbool.h”。