Adapter的软件文件体系

# 一：inc 文件夹

1）inc 文件夹下的所有文件都是头文件，其中包含的子文件夹也是都是各种头文件

2）inc 中的**chip\_15xx\_inc** 是包含了所有和芯片寄存器所有相关的封装设置。

3）inc 中的 adapter\_inc 是包含了适配器的相关类信息

4）inc 中的 driver\_inc 主要是对芯片的各种底层资源进行封装成类，成为资源，负责给上层提供服务。

5） obd\_protocol\_inc 主要是利用底层的硬件资源对各种协议进行实现，为adapter 上层提供服务。

# 二：src 文件夹：

该文件和 inc 文件夹中各个关系是一样的，不过都是包含了各种的源文件。

# 三：chip\_15xx\_inc 中文件：

1. chip.h 该文件定义了**各种和LPC15xx系列**的内存映射宏定义。**各种寄存器组**的基地址宏定义。并且**定义了一些这些地址**对应的**指针**。利用**强制类型转换**成对应的特定寄存器组类型指针，比如 LPC\_UART\_T ，就是与**串口相关的寄存器组**。

2）上述的 **LPC\_UART\_T** 的类型定义其实在 **uart\_15xx.h** ，在该文件中根据数据手册定义了与 UART 相关的寄存器类型，定义了一些基本设置的宏定义，方便代码的易读性。在源文件中自定义了一些简单的函数。

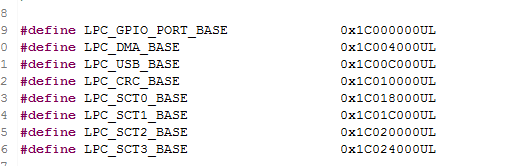
3）对于2）中定义的一些函数，其实在rom 中就已经封装好了一些函数 （API）提供我们使用，这些函数都在rom 的固定位置，通过一个 rom API 表，该表的定义在romapi\_15xx.h 中，该表中是各组不同功能API 表的地址，每组API 通过该表中地址去查表，然后在具体表中查找对应的函数。结构如下：



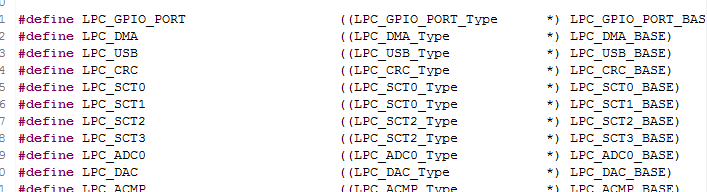
所以我们可以自己定义一组自定义函数，运用各种外设，比如 UART ，Timer，CAN，USB，同时我们也可以用 ROM 中已经定义好的一组API。

# 四：各种文件包含功能

1: chip.h 包含的而是各组寄存器的基地址信息，并且包含了把这些地址转换成对应的操作指针（句柄）的定义：



转换为指针：



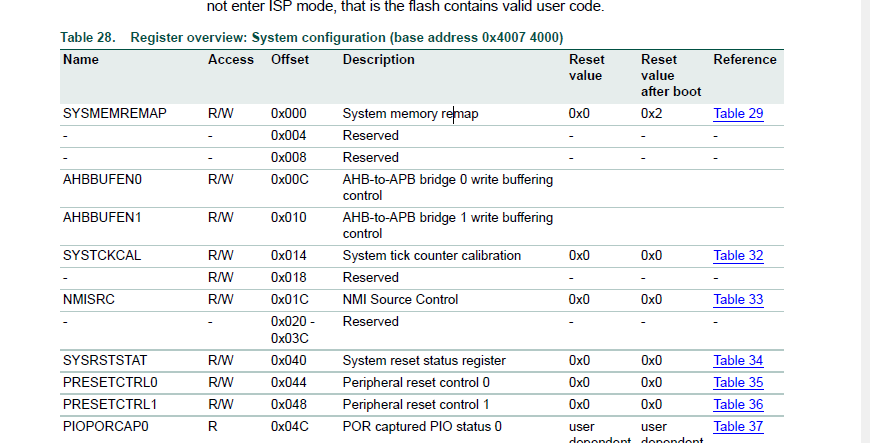
这里先**定义了一个类型 LPC\_GPIO\_PORT\_Typ**e ，然后把 LPC\_GPIO\_PORT\_BASE 转换为对应的类型指针，**那么用 LPC\_GPIO\_PORT 这个指针（宏定义）**就可以去**直接操作对应寄存器**，这里**其实就是为了让编译器**知道**去哪边写数据**，读数据。

对于**LPC\_GPIO\_PORT\_Type 这个类型**，是自己在**其他头文件**中按照**数据手册**定义的。

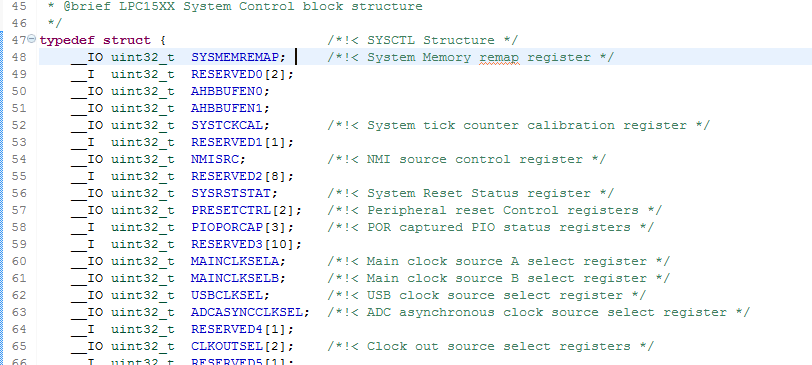
所以**如果想要自己添加相应的功能**，比如**CAN 功能**在标准库中就没有封装，所以**需要添加自己的头文件**，利用数据手册做好封装。然后利用宏定义定义一个对应的指针。

2: **sysctl\_15xx.h**该文件夹下都是与**系统配置寄存器**相关的设置，比

如**时钟频率设置**，时钟源设置。对比如下两幅图片就可以看出来对应的名字都是一样的。

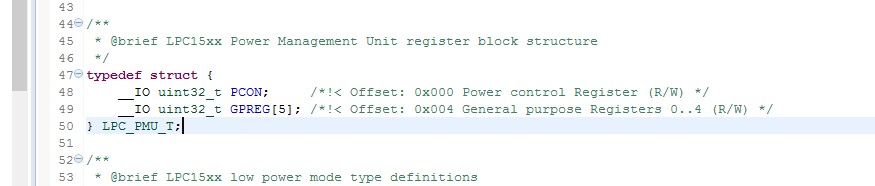


图一

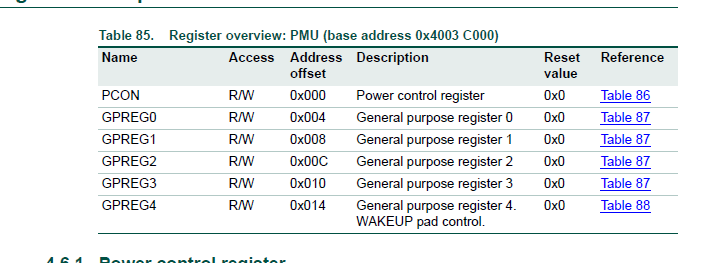


图二

3：pmu\_15xx.h 中的相关设置，是对芯片的电源管理。寄存器定义如下图



图三



图四

4：iocon\_15xx.h 是对IO口的**配置寄存器的底层封装**，类似于前面描述的类型。

5：swm\_15xx.h 是对 Switch Matrix 功能相关寄存器的配置和设置。

6：pmu\_15xx.h 则是对电源管理的寄存器封装。

7：gpio\_15xx.h 则是一个对这些io底层配置完毕后，进行**操作封装**，提供**应用**的文件。为**更上层**提供**硬件层接口**。同时也对**IO口状态寄存器**进行**封装**了。

8：inmux\_15xx.h 是对input mux 寄存器的封装设置。

9：pinint\_15xx.h 是对 pin interrupt 的寄存器封装和相应的宏定义。

10：dma\_15xx.h 是对DMA 的相关设置和封装。

11：sctipu\_15xx.h 是 **SCT Input Processing Unit 设置。**

12：sct\_pwm\_15xx.h 是对应的 SCTimer/PWM 相关的设置。他是基于 sct\_15xx.h

的基础来完成的。

13：sct\_15xx.h 是对 SCT 相关寄存器进行封装了。

14：mrt\_15xx.h 是对Multi-Rate Timer(MRT) 的封装

15：ritimer\_15xx.h 是对 Repetitive Interrupt Timer（RIT）的封装。

16：uart\_15xx.h 是串口相关的。

17：c\_can0\_15xx.h 是本人添加的库文件。其实也可以直接用 rom\_can\_15xx.h 中的 rom API 设置CAN ，但是封装之后可以提供后用。

19：ring\_buffer.h 的文件中定义了一个环形队列。详见：

<https://en.wikipedia.org/wiki/Circular_buffer>