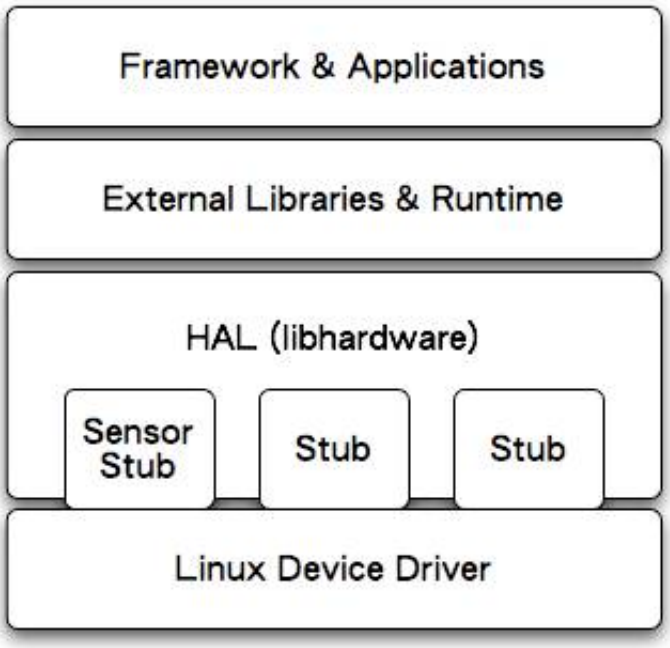
# Android硬件抽象层（HAL）概要介绍和学习计划

Android的硬件抽象层，简单来说，就是对Linux内核驱动程序的封装，向上提供接口，屏蔽低层的实现细节。也就是说，把对硬件的支持分成了两层，一层放在用户空间（User Space），一层放在内核空间（Kernel Space），其中，硬件抽象层运行在用户空间，而Linux内核驱动程序运行在内核空间。为什么要这样安排呢？把硬件抽象层和内核驱动整合在一起放在内核空间不可行吗？从技术实现的角度来看，是可以的，然而从商业的角度来看，把对硬件的支持逻辑都放在内核空间，可能会损害厂家的利益。我们知道，Linux内核源代码版权遵循GNU License，而Android源代码版权遵循Apache License，前者在发布产品时，必须公布源代码，而后者无须发布源代码。如果把对硬件支持的所有代码都放在Linux驱动层，那就意味着发布时要公开驱动程序的源代码，而公开源代码就意味着把硬件的相关参数和实现都公开了，在手机市场竞争激烈的今天，这对厂家来说，损害是非常大的。因此，Android才会想到把对硬件的支持分成硬件抽象层和内核驱动层，内核驱动层只提供简单的访问硬件逻辑，例如读写硬件寄存器的通道，至于从硬件中读到了什么值或者写了什么值到硬件中的逻辑，都放在硬件抽象层中去了，这样就可以把商业秘密隐藏起来了。也正是由于这个分层的原因，Android被踢出了Linux内核主线代码树中。大家想想，Android放在内核空间的驱动程序对硬件的支持是不完整的，把Linux内核移植到别的机器上去时，由于缺乏硬件抽象层的支持，硬件就完全不能用了，这也是为什么说Android是开放系统而不是开源系统的原因。

撇开这些争论，学习Android硬件抽象层，对理解整个Android整个系统，都是极其有用的，因为它从下到上涉及到了Android系统的硬件驱动层、硬件抽象层、运行时库和应用程序框架层等等，下面这个图阐述了硬件抽象层在Android系统中的位置，以及它和其它层的关系：




在学习Android硬件抽象层的过程中，我们将会学习如何在内核空间编写硬件驱动程序、如何在硬件抽象层中添加接口支持访问硬件、如何在系统启动时提供硬件访问服务以及 如何编写JNI使得可以通过Java接口来访问硬件，而作为中间的一个小插曲，我们还将学习一下如何在Android系统中添加一个C可执行程序来访问硬件驱动程序。由于这是一个系统的学习过程，笔者将分成六篇文章来描述每一个学习过程，包括:

一. 在Android内核源代码工程中编写硬件驱动程序。

二. 在Android系统中增加C可执行程序来访问硬件驱动程序。

三. 在Android硬件抽象层增加接口模块访问硬件驱动程序。

四. 在Android系统中编写JNI方法在应用程序框架层提供Java接口访问硬件。

五. 在Android系统的应用程序框架层增加硬件服务接口。

六. 在Android系统中编写APP通过应用程序框架层访问硬件服务。

学习完这六篇文章，相信大家对Android系统就会有一个更深刻的认识了，敬请关注。