实验 12-13 物理地址虚拟地址以及 GPIO 初始化

12.1 本章导读

Linux 系统中对 IO 的操作，专门设计了一套函数，通过这些函数就可以访问 IO，但是这些函数在控制 GPIO 相关寄存器的时候，要用到 MMU 内存管理单元。所以必须先理解内存管理单元，才能理解 Linux 如何控制 GPIO，本实验给大家介绍物理地址虚拟地址就是属于内存管理单元的内容。

12.1.1 工具

12.1.1.1 硬件工具

1）PC 机

12.1.1.2 软件工具

1）虚拟机 Vmware

2）Ubuntu12.04.2

12.1.2 预备课程

无

12.1.3 视频资源

本节配套视频为“视频 12_物理地址虚拟地址”

本节配套视频为“视频 13_GPIO 初始化”

12.2 学习目标

本章需要学习以下内容：

了解物理地址

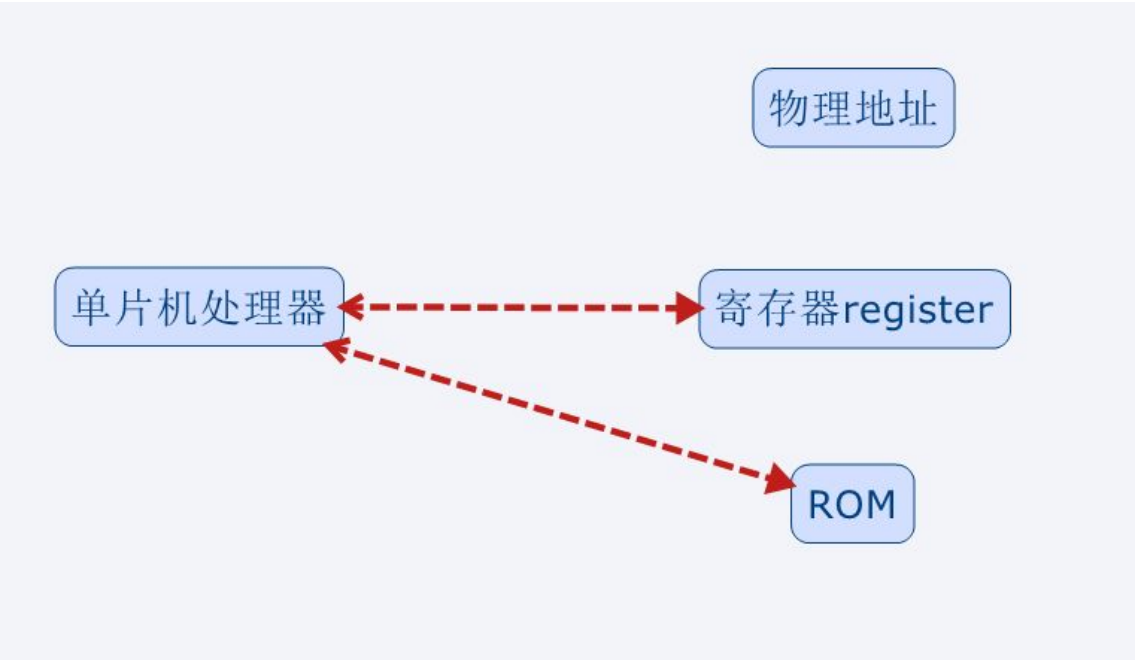
了解虚拟地址

了解内存管理单元

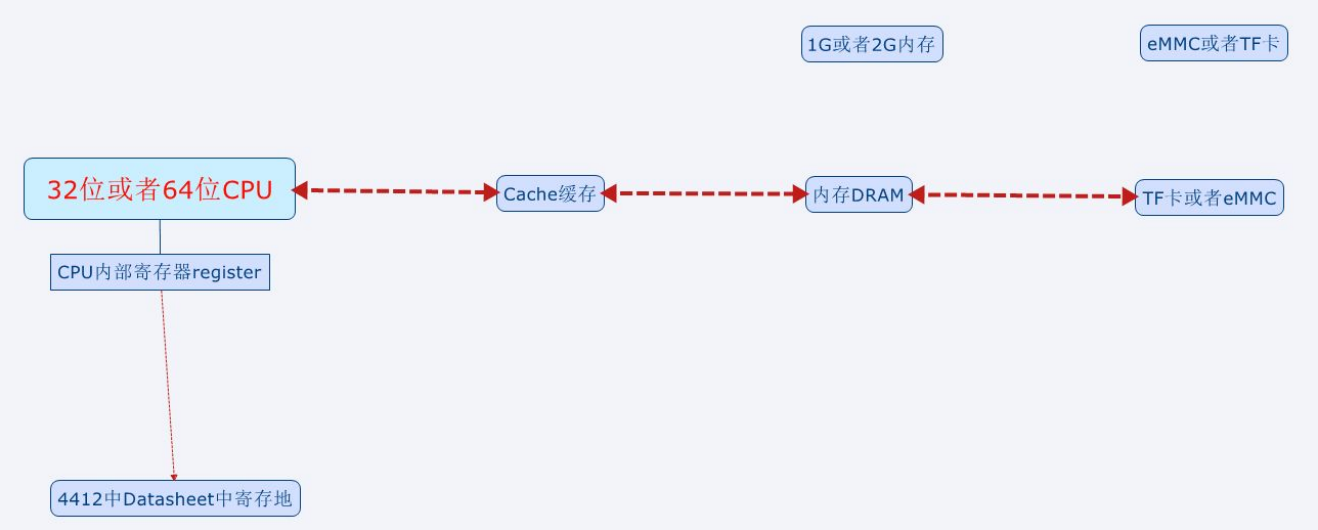
12.3 单片机处理器和现代处理器

在学习单片机的时候，都是可以直接对寄存器和地址操作。

如下图所示，在单片机的世界中，存储器结构比较简单，一般就分为寄存器 Reg 和一个只读存储器 ROM。



如下图所示，是现代中央处理器存储器。其中的 DRAM 就是非常熟悉的内存，32 位处理器一般最大内存是 4G。Cache 是高速缓存 ,用于解决内存和 CPU 之间速度不匹配的问题。ROM 的种类也是繁多，TF 卡、硬盘、eMMC、flash 等等。



如上图所示，其中 CPU 的内部寄存器是和 4412Datasheet 中的寄存器一一对应，在单片机中一般是直接对寄存器进行操作，但是在可以运行带有操作系统的 CPU 中，例如 linux，就不太可能直接进行操作，首先是会对地址进行宏定义，只需要调用已经写好的函数对这些宏定义进行操作，就相当于操作寄存器。

12.4 MMU 内存管理单元

如果理解了本节的内存管理单元，那么前面很多疑惑的地方都可以明白。

MMU 是中央处理用来管理虚拟内存、物理存储器的控制线路，同时将虚拟地址映射为物理地址。

先介绍一下 MMU 的由来。

在最初的 PC 上，内存都非常小，如果程序小于内存，也还是可以运行的。

随着应用程序的出现，出现了一个程序占用的空间大于内存，例如 1M 内存的 PC，想要运行一个 4M 的程序，那么就有一个难题摆在面前了。程序员通常把程序分割为很多个小块，然后一个小块一个小块的运行，虽然调用是有操作系统解决的，但是分割却必须由程序完成，这个过程费时费力，很无聊。人们找到了一个解决办法，那就是虚拟存储器。

虚拟存储器的基本思想是程序，数据，堆栈的总的大小可以超过物理存储器的大小，操作系统把当前使用的部分保留在内存中，而把其他未被使用的部分保存在磁盘上。比如对一个 16MB 的程序和一个内存只有 4MB 的机器，操作系统通过选择，可以决定各个时刻将哪 4M 的内容保留在内存中，并在需要时在内存和磁盘间交换程序片段，这样就可以把这个 16M 的程序运行在一个只具有 4M 内存机器上了，而这个 16M 的程序在运行前不必由程序员进行分割。（如果感兴趣可以找关于现代操作系统的书来看看）。

12.5 物理地址虚拟地址以及 GPIO 的初始化

这部分内容可以参考配套的视频来学习。或者通过网络上的资料来理解这几个知识点。