1. 專案21\_I2C:

* I2C是一種很常見的同步，串列通訊接口，通常用於連接各式各樣的傳感器或觸控螢幕。在Linux系統中，為了方便驅動開發人員，I2C也導入了類似platform的總線驅動框架，但不同於platform的是I2C是實體總線(bus)。

1. Linux底下I2C總線框架:

* Linux內核開發者為了讓驅動開發工程師在內核中方便地添加自己的I2C設備驅動程序，方便大家更容易地在Linux下驅動自己的I2C接口硬件，進而引入了I2C總線框架，包含以下內容:
  1. I2C核心 (I2C-core) I2C核心提供了I2C總線驅動（適配器）和設備驅動的註冊、註銷方法，也提供I2C通信方法與具體硬件無關的代碼以及探測設備地址的上層代碼等(Linux Kernel已經完成)。
  2. I2C總線驅動 (I2C adapter) I2C總線驅動是I2C適配器的軟件實現，提供I2C適配器與從設備間完成數據通信的能力。I2C總線驅動由i2c\_adapter和i2c\_algorithm來描述。I2C適配器是SoC中內置I2C控制器的軟體抽象，可以理解為它所代表的是一個I2C主機。

(由SOC晶片廠商提供)

* 1. I2C設備驅動(I2C client driver)包括兩部分：設備的註冊和驅動的註冊。I2C子系統幫助內核統一管理I2C設備，讓驅動開發工程師在Kernel中可以更加容易地添加自己的I2C設備驅動程序。

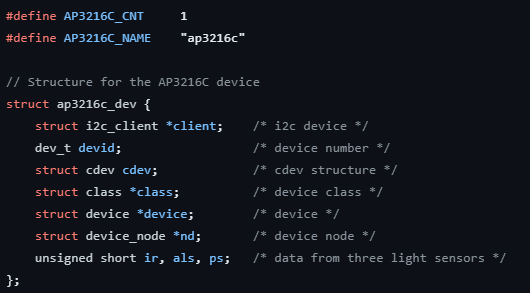
1. I2C總線驅動:

* I2C總線驅動重點是I2C適配器（也就是SoC的I2C接口控制器）驅動，這裡要用到兩個重要的數據結構：i2c\_adapter和i2c\_algorithm。I2C子系統將SoC的I2C適配器（控制器）抽象成一個i2c\_adapter結構體，i2c\_adapter結構體定義在include/linux/i2c.h文件中
* 對於一個I2C適配器，肯定要對外提供讀寫API函數，設備驅動程序可以使用這些API函數來完成讀寫操作。i2c\_algorithm就是I2C適配器與外部I2C設備進行通信的方法。

1. I2C總線設備:

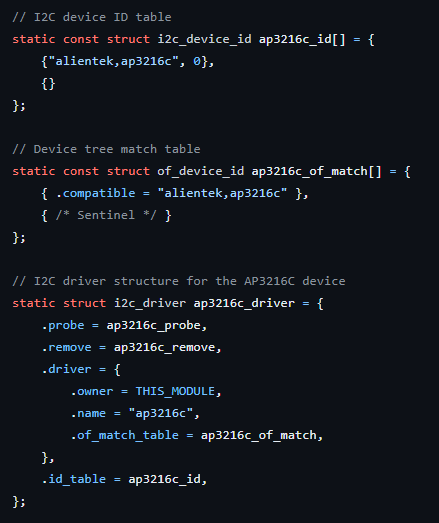
* I2C設備驅動重點關注兩個數據結構：i2c\_client和i2c\_driver。根據總線、設備和驅動模型，i2c\_client用於描述I2C總線下的設備，相當於，platform總線下的platform\_device，而i2c\_driver則用於描述I2C總線下的設備驅動，類似於platform總線下的platform\_driver。

1. I2C驅動實作:
   1. 首先定義設備結構體，宣告struct i2c\_client成員變數。(如下圖一)



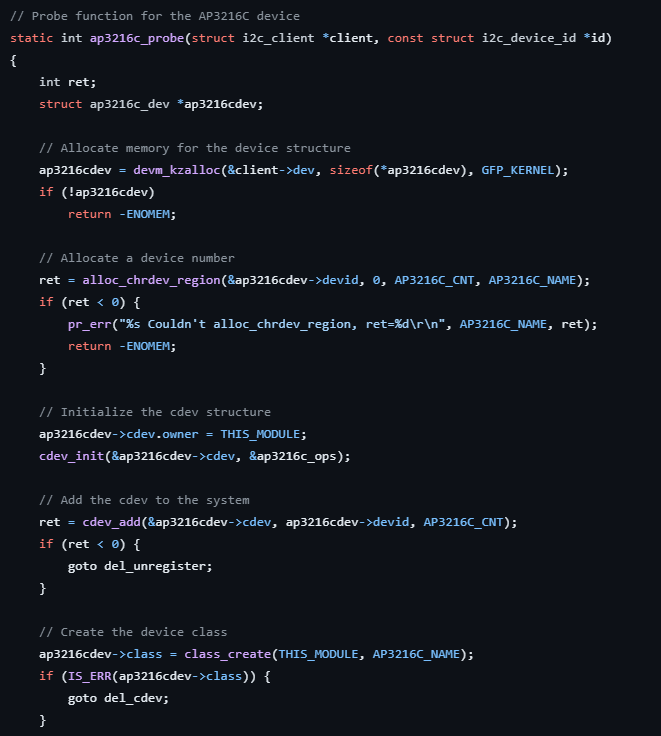
(圖一:設備結構體內容)

* 1. 接著定義，id\_table匹配設備和驅動(i2c\_device\_id確保驅動程序可以在不使用Device Tree的系統上也正常工作。)(如下圖二)



(圖二: id\_table結構體內容)

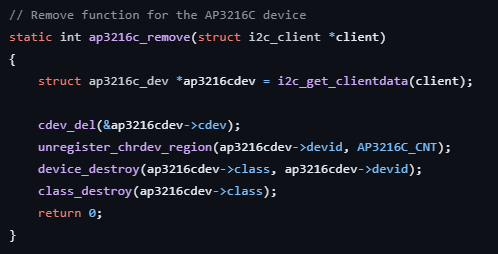
* 1. 實現probe()及remove()函數(如下圖三，圖四)。值得注意的是，在probe()函是透過i2c\_set\_clientdata()來設置private data，並讓remove()函式透過i2c\_get\_clientdata()來取得private data。



(圖三: probe()函式具體實現)

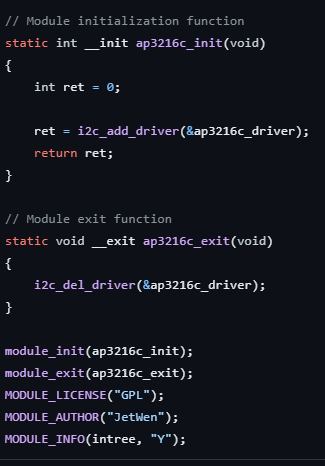


(圖四:probe()函式具體實現)



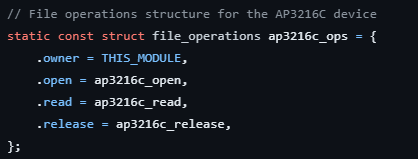
(圖五remove()函式具體實現)

* 1. 驅動出/入口函數實現透過i2c\_add\_driver()函數和i2c\_del\_driver()函數，向Linux Kernel註冊和註銷I2C驅動程式接口(如下圖六)。剩餘部分就是關於本驅動的相關訊息。



(圖六:驅動出/入口函數內容實現)

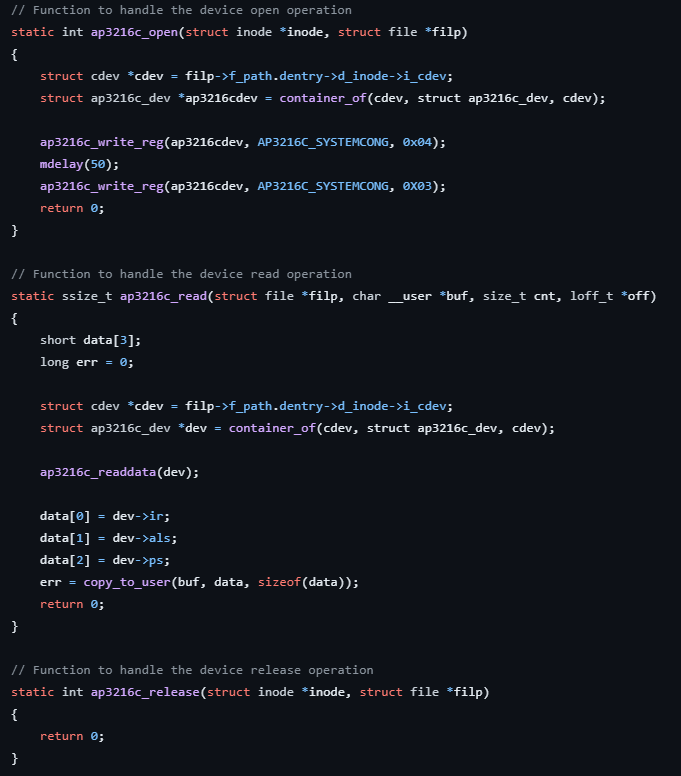
* 1. 接著是定義file\_operations結構體操作函數(如下圖七)。並實現open()，read()，release()函式(如下圖八)。



(圖七: file\_operations結構體內容)

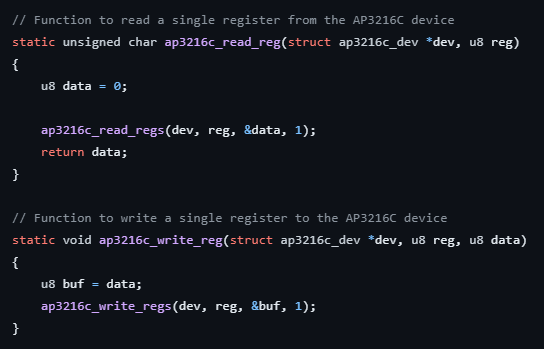
* 1. 注意到open()函式內有ap3216c\_write\_reg()函式用來寫入資料到設備暫存器內部。而ap3216c\_readdata()函式用來讀取設備暫存器內部資料

(如下圖八)



(圖八:open()，release()，read()函式內容實現)

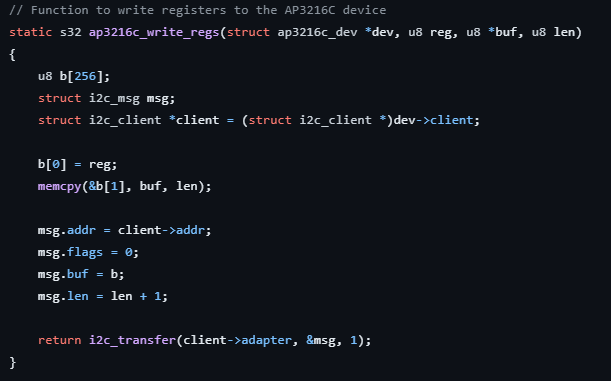
* 1. 最後完善，ap3216c\_write\_reg()函式和ap3216c\_readdata()函式。簡單來說，就是操作設備暫存器來讀取/寫入資料。
* 先編寫讀取一個暫存器的值的函式ap3216c\_read\_reg()(如下圖九)，在透過呼叫ap3216c\_read\_reg()來讀取多個暫存器的值，並將其定義成函式ap3216c\_read\_regs()(如下圖十)。
* 先編寫寫入一個暫存器的值的函式ap3216c\_write\_reg()(如下圖九)，在透過呼叫ap3216c\_write\_reg()來寫入多個暫存器的值，並將其定義成函式ap3216c\_write\_regs()(如下圖十一)。



(圖九: ap3216c\_read\_reg()和ap3216c\_write\_reg()函數實現)



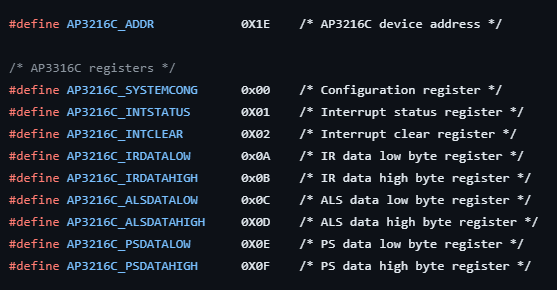
(圖十: ap3216c\_read\_regs()函數實現)



(圖十一: ap3216c\_write\_regs()函數實現)

* 1. 此外我們還將暫存器地指按照DataSheet定義在一個header file中。

(如下圖十二)



(圖十二:設備暫存器地址定義)

※總結:

整體而言I2C已經在Linux系統中，被定義好了而且是透過總線框架方式完成，所以我們只需要完善整體框架就可以了。也因為這樣的設計讓Linux系統做到完整的驅動分離和分層。對於程式開發人員來說編寫起來相對輕鬆許多。

接著I2C讀取和寫入暫存器操作就跟一般Bare-Metal驅動操作類似，按照設備的datasheet去寫就可以了。