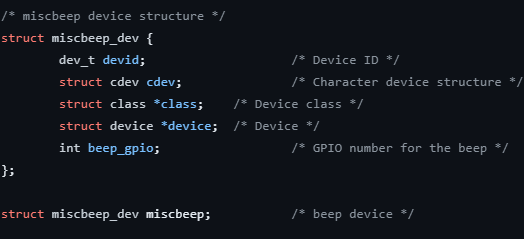
1. 專案19\_misc:

* MISC（Miscellaneous）設備是一種在Linux系統中使用的設備類型，用於處理那些不適合歸類到標準設備類型（如塊設備、字符設備、網絡設備等）中的設備。MISC設備的概念和機制主要是為了簡化這些“雜項”設備的管理和開發。但是其本質上就是最單純的字元設備驅動嵌套在platform驅動框架中。

1. MISC設備概述:

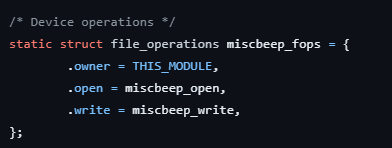
* 隨著Linux字元設備不斷增加，為避免主設備號不夠使用，所以MISC設備會自動創建cdev，因此MISC設備驅動可以簡化字元設備驅動的編寫。我們需要向 Linux 註冊一個 miscdevice 設備，miscdevice 是一個結構體，定義在文件 include/linux/miscdevice.h。

1. MISC實作:
   1. 首先定義設備結構體。(如下圖一)



(圖一:設備結構體內容)

* 1. 接著定義file\_operations結構體(如下圖二)並實現其中操作函數open()和write()。(如下圖三)

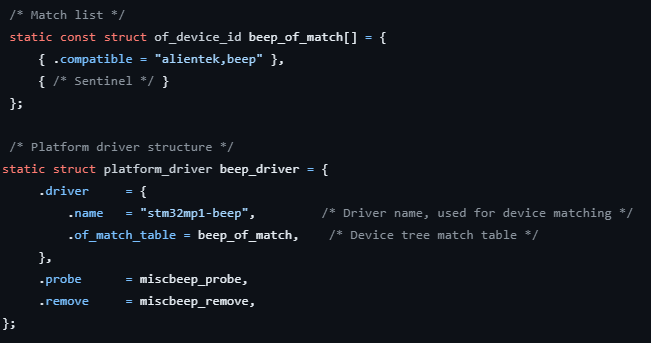


(圖二: file\_operations結構體內容)



(圖三:open()函式和write()函式內容)

* 1. 再來定義id\_table，在platform驅動框架中透過Device Tree來匹配驅動和設備(如下圖四)，並實現platform\_driver結構體中probe()函式和remove()函式內容。



(圖四:id\_table和platform\_driver結構體內容)

* 1. probe()函式的實現，先透過beep\_gpio\_init()函式來初始化beep，再透過misc\_register()函式註冊為MISC設備。(如下圖五)



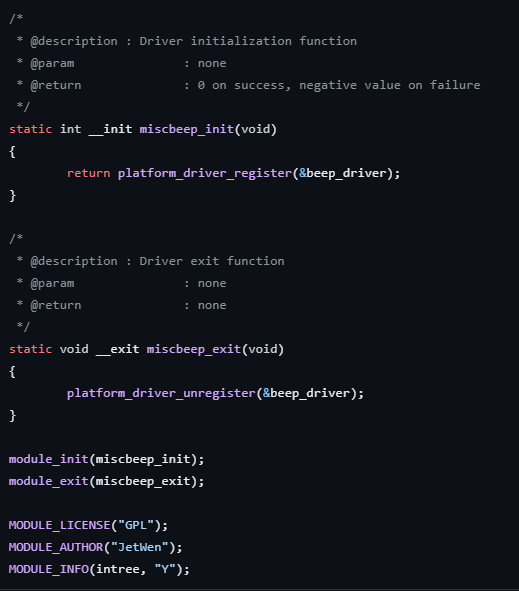
(圖五:probe()函式內容)

* 1. 接著是remove()函式的實現，比重要的部分是透過misc\_deregister()函式來註銷MISC設備。(如下圖六)



(圖六:remove()函數內容實現)

* 1. 最後一步就是，驅動出/入口函數，以及本驅動相關訊息的描述，特別注意，在驅動入口函數中透過platform\_driver\_register()函數來向Kernel註冊platform\_driver，最後在驅動出口函數中透過platform\_driver\_unregister()函數來向Kernel註銷platform\_driver。(如下圖七)



(圖七:驅動出/入口函式內容實現)

※總結:

當需要編寫字元設備驅動且透過platform框架來完善時，MISC設備驅動是個不錯的選擇可以幫助程式碼模塊化，也可以使程式碼更加簡潔，更加容易理解。