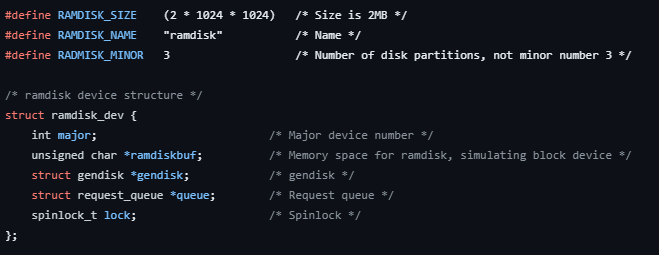
1. 專案27\_ramdisk\_norequest

* 這個專案將模擬RAM磁碟的Linux塊設備驅動程序，不使用request的條件下，實作Linux塊設備驅動。

1. ramdisk\_norequest實作:
2. 首先，定義設備結構體，來儲存設備相關訊息(如下表一，圖一)。

(表一: 設備結構體成員變數內容)

|  |  |  |
| --- | --- | --- |
| **成員名稱** | **類型** | **功能描述** |
| major | int | 主要設備號碼，用於標識設備。 |
| ramdiskbuf | unsigned char \* | 指向 RAM 磁碟內存區域的指標，用於模擬塊設備的存儲空間。 |
| gendisk | struct gendisk \* | 指向 gendisk 結構體的指標，gendisk 是內核中表示塊設備的數據結構，包含了設備名稱、容量等信息。 |
| queue | struct request\_queue \* | 指向請求佇列結構體的指標，請求佇列用於管理 I/O 請求。 |
| lock | spinlock\_t | 自旋鎖，用於保護共享資源，確保並發訪問的同步。 |

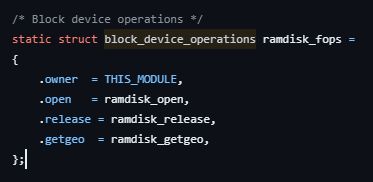


(圖一: 設備結構體成員變數內容)

1. 定義block\_device\_operations結構體成員函式，用於指定塊設備的操作函數(如表二，圖二)。

(表二: block\_device\_operations結構體成員函式)

|  |  |  |
| --- | --- | --- |
| **成員名稱** | **類型** | **功能描述** |
| owner | struct module \* | 指向擁有這個結構體的內核模塊。通常設置為 THIS\_MODULE，表示當前模塊。 |
| open | int (\*)(struct block\_device \*, fmode\_t) | 指向一個函數指標，當打開設備時被調用。這裡設置為 ramdisk\_open。 |
| release | void (\*)(struct gendisk \*, fmode\_t) | 指向一個函數指標，當釋放設備時被調用。這裡設置為 ramdisk\_release。 |
| getgeo | int (\*)(struct block\_device \*, struct hd\_geometry \*) | 指向一個函數指標，當獲取設備幾何結構時被調用。這裡設置為 ramdisk\_getgeo。 |

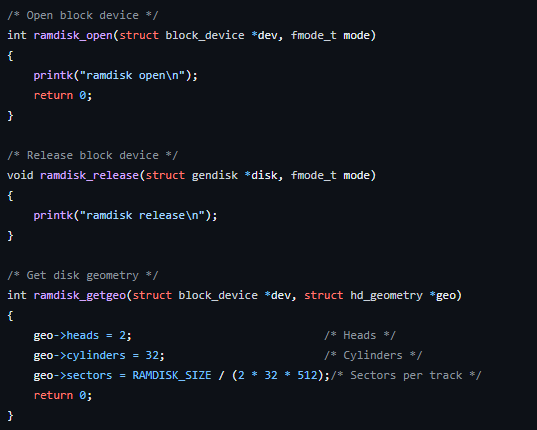


(圖二: block\_device\_operations結構體成員函式)

1. 實現block\_device\_operations中的每一個操作函式。(如下表三，圖三)

(表三: block\_device\_operations中操作函式的實現)

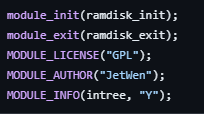
|  |  |  |  |
| --- | --- | --- | --- |
| **函數名稱** | **返回類型** | **參數** | **功能描述** |
| ramdisk\_open | int | struct block\_device \*dev, fmode\_t mode | 打開塊設備，打印開啟信息。 |
| ramdisk\_release | void | struct gendisk \*disk, fmode\_t mode | 釋放塊設備，打印釋放信息。 |
| ramdisk\_getgeo | int | struct block\_device \*dev,  struct hd\_geometry \*geo | 獲取設備幾何結構，返回磁頭數量、柱面數量和每軌道的扇區數量。 |



(圖三: block\_device\_operations中操作函式的實現)

1. 定義驅動出入口函數，及驅動出入口函數實現(如圖四)

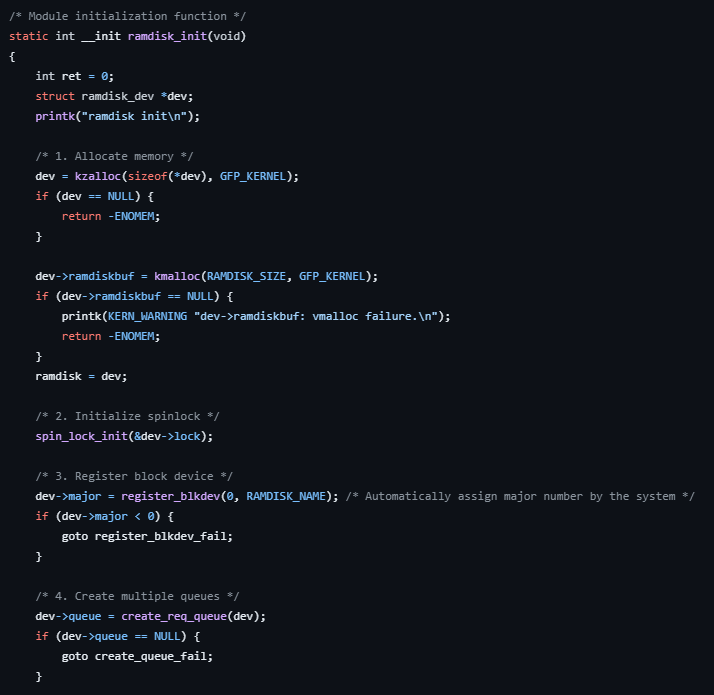
* 驅動入口函數(ramdisk\_init): 該函數在模塊加載時被調用，用於初始化 RAM 磁碟設備。(如下表四，圖五，圖六)



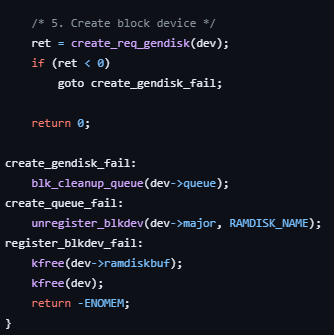
(圖四:驅動出入口函式定義)

(表四: 驅動入口函數(ramdisk\_init)內容解析)

|  |  |
| --- | --- |
| **步驟** | **說明** |
| **打印初始化信息** | 使用 printk 函數打印 "ramdisk init" 信息，用於調試或日志記錄。 |
| **分配內存** | 使用 kzalloc 函數為 ramdisk\_dev 結構體分配內存。 |
| **分配 RAM 磁碟內存區域** | 使用 kmalloc 函數為 RAM 磁碟分配內存區域，如果分配失敗則打印警告信息並返回錯誤。 |
| **初始化自旋鎖** | 使用 spin\_lock\_init 函數初始化自旋鎖，用於保護共享資源，確保並發訪問的同步。 |
| **註冊塊設備** | 使用 register\_blkdev 函數向內核註冊塊設備，並自動分配主設備號。如果註冊失敗則釋放資源並返回錯誤。 |
| **創建請求佇列** | 調用 create\_req\_queue 函數創建請求佇列，如果創建失敗則釋放資源並返回錯誤。 |
| **創建塊設備** | 調用 create\_req\_gendisk 函數創建並初始化 gendisk 結構，然後註冊磁碟。如果創建失敗則釋放資源並返回錯誤。 |



(圖五:驅動入口函數(ramdisk\_init)內容實現)

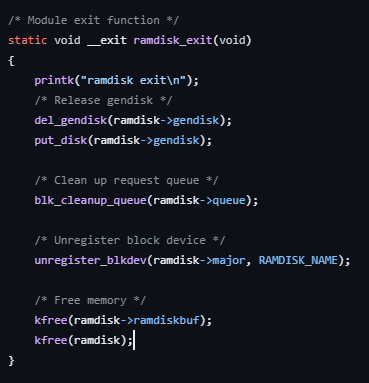


(圖六:驅動入口函數(ramdisk\_init)內容實現)

* 驅動出口函數(ramdisk\_exit):該函數在模塊卸載時被調用，用於釋放 RAM 磁碟設備的資源。(如下表五，圖七)

(表五: 驅動出口函數(ramdisk\_exit)內容解析)

|  |  |
| --- | --- |
| **步驟** | **說明** |
| **打印退出信息** | 使用 printk 函數打印 "ramdisk exit" 信息，用於調試或日志記錄。 |
| **釋放 gendisk** | 使用 del\_gendisk 和 put\_disk 函數釋放 gendisk 結構體。 |
| **清理請求佇列** | 使用 blk\_cleanup\_queue 函數清理請求佇列。 |
| **註銷塊設備** | 使用 unregister\_blkdev 函數註銷塊設備。 |
| **釋放內存** | 使用 kfree 函數釋放 RAM 磁碟的內存區域和 ramdisk\_dev 結構體。 |

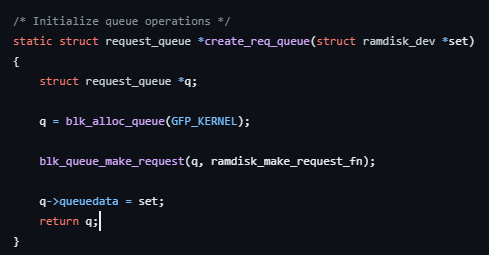


(圖七: 驅動出口函數(ramdisk\_exit)內容實現)

1. 最後實現ramdisk\_init中會用到的create\_req\_gendisk，create\_req\_queue及ramdisk\_make\_request\_fn函式。
   * + create\_req\_queue: 用於創建和初始化請求佇列。(如下表六，圖八)

(表六:create\_req\_queue內容解析)

|  |  |
| --- | --- |
| **步驟** | **說明** |
| **分配請求佇列** | 使用 blk\_alloc\_queue 函數分配請求佇列。 |
| **設置請求處理函數** | 使用 blk\_queue\_make\_request 函數設置請求處理函數。 |
| **設置佇列數據** | 設置佇列的數據指針為設備結構體。 |
| **返回請求佇列** | 返回創建和初始化的請求佇列。 |
|  |  |

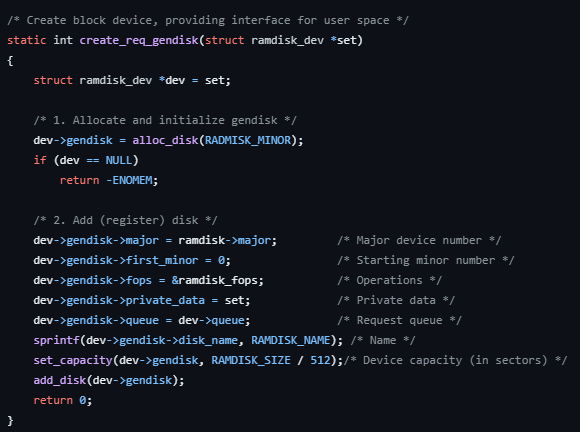


(圖八:create\_req\_queue內容實作)

* + - create\_req\_gendisk: 用於創建和初始化 gendisk 結構，並註冊塊設備，使其在User Space可用。(如下表七，圖九)

(表七: create\_req\_gendisk內容解析)

|  |  |
| --- | --- |
| **步驟** | **說明** |
| **分配和初始化 gendisk** | 使用 alloc\_disk 函數分配並初始化 gendisk 結構。 |
| **檢查分配結果** | 檢查 gendisk 分配是否成功，如果失敗則返回錯誤。 |
| **設置 gendisk 屬性** | 設置 gendisk 的主要屬性，包括主設備號、次設備號、操作函數、私有數據、請求佇列和設備名稱。 |
| **設置設備容量** | 使用 set\_capacity 函數設置設備容量（以扇區為單位）。 |
| **註冊磁碟** | 使用 add\_disk 函數註冊磁碟，使其在系統中可用。 |
| **返回成功** | 所有步驟成功執行後，返回 0。 |



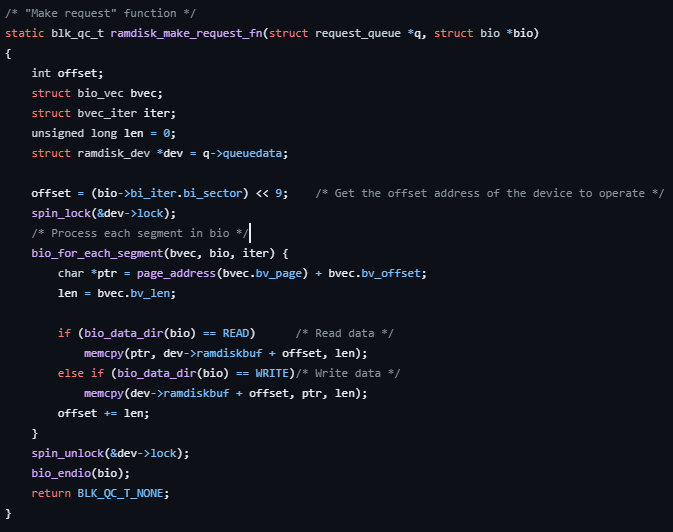
(圖九: create\_req\_gendisk函式內容實作)

* + - ramdisk\_make\_request\_fn: 用於處理塊設備的 I/O 請求。

(如下表八，圖十)

(表八: ramdisk\_make\_request\_fn內容解析)

|  |  |
| --- | --- |
| **步驟** | **說明** |
| **計算偏移地址** | 計算設備要操作的偏移地址。 |
| **加鎖** | 使用自旋鎖保護共享資源，確保並發訪問的同步。 |
| **處理每個 bio 段** | 遍歷 bio 中的每個段，進行讀寫操作。 |
| **讀取或寫入數據** | 根據 bio 的方向進行讀取或寫入操作。 |
| **釋放鎖** | 釋放自旋鎖。 |
| **完成 bio** | 標記 bio 操作完成。 |
| **返回** | 返回 BLK\_QC\_T\_NONE。 |



(圖十: ramdisk\_make\_request\_fn內容實作)

※總結:

這段專案實做了一個模擬的 RAM 磁碟 Linux 驅動程序，該驅動程序允許系統將 RAM 的一部分作為塊設備進行操作。驅動程序包括初始化和退出函數、打開和釋放設備的函數、獲取磁碟幾何結構的函數、處理 I/O 請求的函數等。

(具體流程步驟參考下表九。)

(表九:整體Block Device in Linux Kernel驅動開發流程及步驟詳解)

|  |  |
| --- | --- |
| **步驟** | **說明** |
| **包含必要的標頭文件** | 包含各種內核模塊開發所需的標頭文件。 |
| **定義常量和數據結構** | 定義 RAMDISK 的大小、名稱、磁碟分區數量，並定義 ramdisk\_dev 結構體。 |
| **定義設備操作函數** | 定義打開、釋放和獲取磁碟幾何結構的函數。 |
| **定義塊設備操作結構體** | 定義 block\_device\_operations 結構體，設置各種操作函數。 |
| **處理 I/O 請求函數** | 定義 ramdisk\_make\_request\_fn 函數，處理讀寫請求。 |
| **初始化佇列操作** | 定義 create\_req\_queue 函數，創建並初始化請求佇列。 |
| **創建塊設備** | 定義 create\_req\_gendisk 函數，創建並初始化 gendisk 結構，並註冊磁碟。 |
| **模塊初始化函數** | 定義 ramdisk\_init 函數，初始化模塊，包括分配內存、初始化自旋鎖、註冊塊設備、創建請求佇列和創建塊設備。 |
| **模塊退出函數** | 定義 ramdisk\_exit 函數，在模塊卸載時釋放資源，包括刪除 gendisk、清理請求佇列、註銷塊設備並釋放內存。 |
| **模塊入口和出口** | 使用 module\_init 和 module\_exit 宏定義模塊的入口和出口函數。 |