Project 2: CPU Synchronous Virtual Device

by 黃謙仁，林群崴，傅家靖，黃懋翔，胡程維，高新造

設計：

本次實驗使用 Linux Kernel 5.0.10 搭配 Ubuntu 19.04 OS System。利用 synchronous file I/O 及 memory-mapped I/O 兩種方式在 master-slave model 進行模擬並比較兩者的 performance。mmap(memory-mapped I/O)將檔案內容映射到記憶體，file descript 當作起始位址，檔案大小則是映射長度，通過對記憶體的讀取，實現利用虛擬位址存取檔案。

以下分成兩端分別敘述：

Master 端

在 Master 端 initialize 後，master 端會得知檔案大小在傳輸之前，並將完整檔案映射到虛擬位址上，而 master device 會透過虛擬映射實體位址存取檔案內容。其中使用 mmap 節省 file I/O 需要花費的 buffer 時間，進而減少原本從 user 端 copy 檔案過來的時間。

Slave 端

Slave 端不會得知檔案的大部分資訊，全由 Mster 端進行掌握，因此映射時先開啟一給定大小位址的 buffer，逐漸根據傳來的檔案進行調整，buffer 滿了就另開新檔，直到整個檔案從 user 端傳輸完成。跟 Master 端不同的是，檔案本身為空，為了不讓這空檔案開始映射實體位址讓錯誤產生，每次在開啟新的映射位址時，該檔案的對應 offset 位置必須寫入一 null character，再於 transmission 完成後將不必要的暫用 offset 移除，以保證 Slave 端不會報錯。

整個架構中，Master 端獲得的授權較多，而 Slave 端的授權則優先權低於 Master，這樣的不平行使得在效能上比起完全平行執行的架構較差，但是會具有更多的安全性和控制性，只有在 Master 端中才能快速映射到實體的記憶體位址，並且可以調控 Slave 執行的順序、時間等；當然在 slave 端也可以使用 mmap，但是檔案較大的時候，時間明顯會相較 Master 慢很多。

執行範例測資的結果：

mmap

```
qunwei@qunwei-X555LB:~/os_project2_2018$ sh exec.sh m
./data/5.in
ioctl success
Master transmission time: 0.033600 ms, File size: 625 bytes with method mmap
0.033600, 625, mmap
Slave transmission time: 0.038800 ms, File size: 625 bytes with method mmap
0.038800, 625, mmap
./data/20.in
ioctl success
Master transmission time: 0.040600 ms, File size: 2500 bytes with method mmap
0.040600, 2500, mmap
Slave transmission time: 0.056500 ms, File size: 2500 bytes with method mmap
0.056500, 2500, mmap
./data/4000.in
ioctl success
Master transmission time: 1.114600 ms, File size: 500000 bytes with method mmap
1.114600, 500000, mmap
Slave transmission time: 2.436300 ms, File size: 500000 bytes with method mmap
2.436300, 500000, mmap
./data/300.in
ioctl success
Master transmission time: 0.088900 ms, File size: 37500 bytes with method mmap
0.088900, 37500, mmap
Slave transmission time: 0.246600 ms, File size: 37500 bytes with method mmap
0.246600, 37500, mmap
./data/1.in
ioctl success
Master transmission time: 0.095300 ms, File size: 125 bytes with method mmap
0.095300, 125, mmap
Slave transmission time: 0.038100 ms, File size: 125 bytes with method mmap
0.038100, 125, mmap
qunwei@qunwei-X555LB:~/os_project2_2018$
```

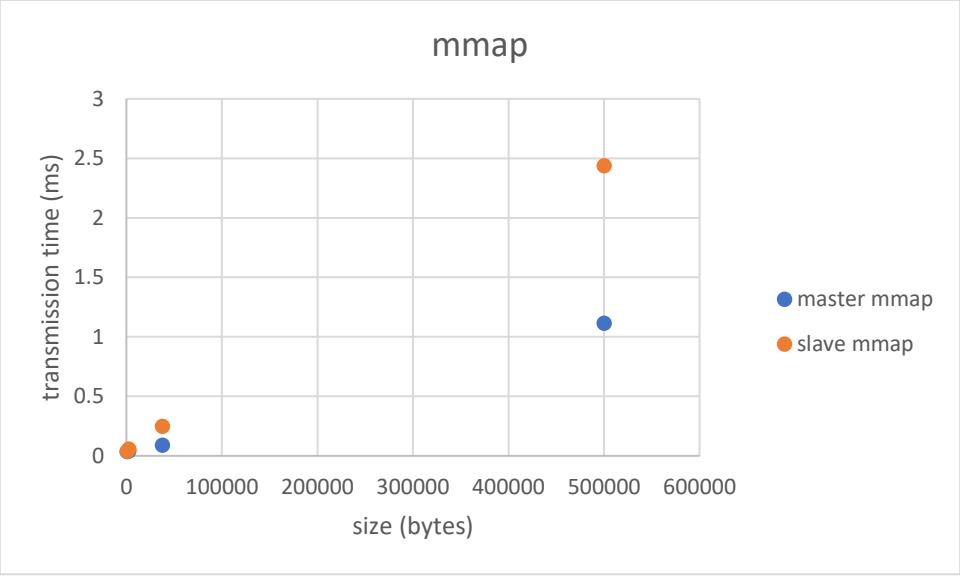
file

```
qunwei@qunwei-X555LB:~/os_project2_2018$ sh exec.sh f
./data/5.in
ioctl success
Master transmission time: 0.018500 ms, File size: 625 bytes with method fcntl
0.018500, 625, fcntl
Slave transmission time: 0.055200 ms, File size: 625 bytes with method fcntl
0.055200, 625, fcntl
./data/20.in
ioctl success
Master transmission time: 0.252000 ms, File size: 2500 bytes with method fcntl
0.252000, 2500, fcntl
Slave transmission time: 0.053700 ms, File size: 2500 bytes with method fcntl
0.053700, 2500, fcntl
./data/4000.in
ioctl success
Master transmission time: 1.697800 ms, File size: 500000 bytes with method fcntl
1.697800, 500000, fcntl
Slave transmission time: 2.313300 ms, File size: 500000 bytes with method fcntl
2.313300, 500000, fcntl
./data/300.in
ioctl success
Master transmission time: 0.118000 ms, File size: 37500 bytes with method fcntl
0.118000, 37500, fcntl
Slave transmission time: 0.228000 ms, File size: 37500 bytes with method fcntl
0.228000, 37500, fcntl
./data/1.in
ioctl success
Master transmission time: 0.055900 ms, File size: 125 bytes with method fcntl
Slave transmission time: 0.026800 ms, File size: 125 bytes with method fcntl
0.055900, 125, fcntl
0.026800, 125, fcntl
qunwei@qunwei-X555LB:~/os_project2_2018$
```

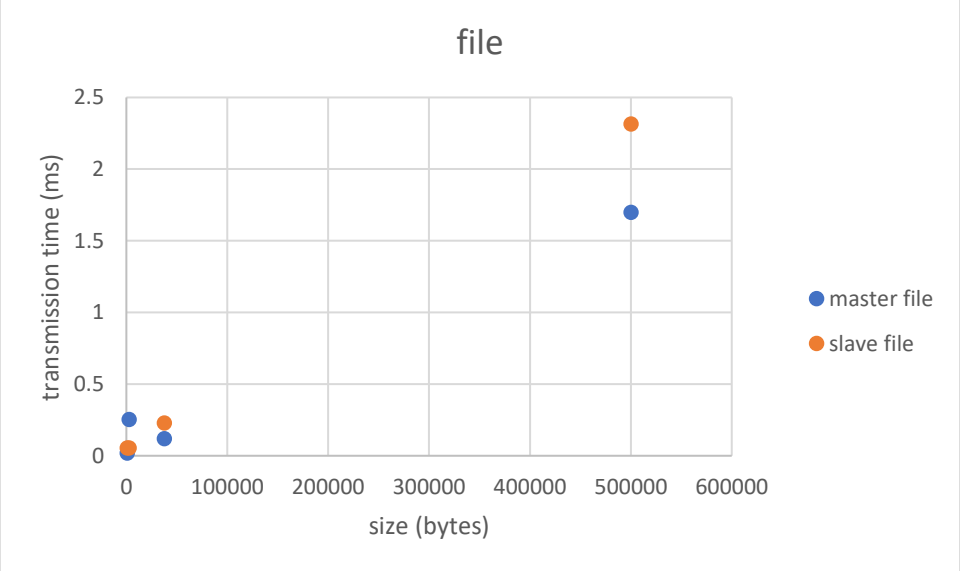
Size (bytes)	master mmap	slave mmap	master file	slave file
125	0.0953	0.0381	0.0559	0.0268
625	0.0336	0.0388	0.0185	0.0522
2500	0.0406	0.0565	0.2520	0.0537
37500	0.0889	0.2466	0.1180	0.2280
500000	1.1146	2.4363	1.6978	2.3133 (ms)

比較實際結果與理論結果：

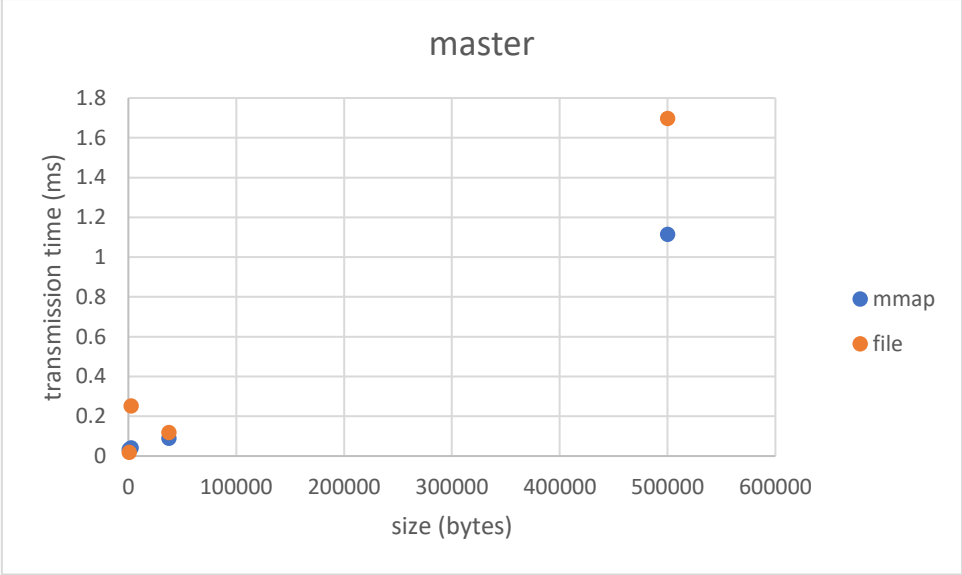
在上方表格，可發現在檔案大小 125 bytes，會有較大不合理的 transmission time，以下圖皆不繪製 125 bytes 的情況。



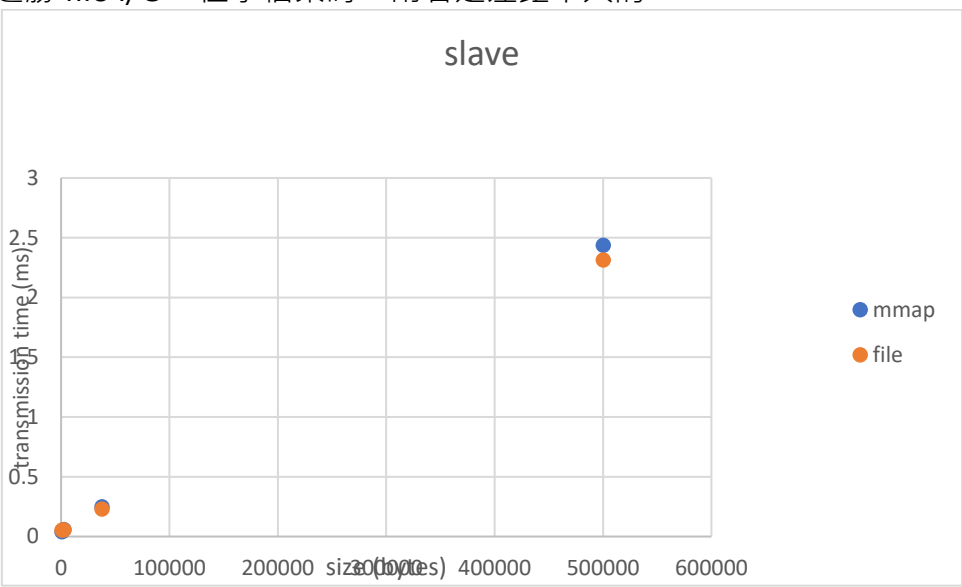
從圖可得知，檔案越大，slave 端與 master 端花費的時間差距更加明顯，與 slave 端在 transmission 需要加大虛擬位址的情況相符合。



(忽略 master 一不合理的數據)從圖得知檔案越大，slave 端與 master 端花費的時間差距更加明顯，但沒有如同 mmap 那樣明顯。



(忽略 file I/O 一不合理的數據)從圖可驗證上一圖的理論，大檔案時 mmap 的效率遠勝 file I/O，但小檔案時，兩者是差距不大的。



因 slave 端在 transmission 需要加大虛擬位址，mmap 和 file I/O 的 performance 差距較小。

依據上四圖，因為 mmap 的初始化需要將檔案內容映射到記憶體，花費較大，檔案小的時候 file I/O performance 稍微高一點，但大檔案時 mmap performance 效率明顯更佳。

各組員的貢獻：

黃謙仁：製造實驗數據並分析實驗結果。
林群崴：實作撰寫 mmap I/O 並實際進行測試。
傅家靖：實作撰寫 user program 並實際進行測試。
黃懋翔：分析程式設計並討論理論與程式結果的差異。
胡程維：統計實驗結果與呈現資料。
高新造：反覆測試並檢查程式正確性。