**OS Project 2 Report--Group 29**

**Design** 這是一個Master-Slave架構，我們需要讓Master以及Slave device都支援mmap。以下分別說明四份程式碼:

user\_program/master.c:  
將檔案map到user\_program記憶體後，再將device記憶體map到user\_program，而後利用memcpy將檔案拷貝至該map記憶體，再透過ioctl通知device mapping已完成。

master\_device/master\_device.c:

抓到user\_program中master.c傳送的通知後，找到記憶體資料，再透過ksend傳送記憶體資料給slave device。

user\_program/slave.c:

Slave device讀到檔案後，slave program使用mmap以及memcpy寫道output file，傳page的address去printk

slave\_device/slave\_device.c:

連線後取得資料，將device處記憶體map至user\_program，再將內容map到將輸出的檔案

**Test Case:**

**Result analysis and compare file I/O & memory-mapped I/O:**   
若是小檔案如file1\_in、file2\_in，執行的時間小於0.1ms時，m/m並沒有明顯的優勢，但隨著檔案越來越大，兩者逐漸拉開的差距，說明了mmap I/O在大檔案時，確實能夠提高傳遞效率。

Master side:  
檔案漸大時，Master方使用會比使用fcntl快。

Reason:  
mmap時呼叫system call的次數較少。

Slave side:  
Slave方使用mmap與fcntl的時間快慢無固定。

Reason:  
接收檔案無法事先得知大小，導致得事後壓縮，因此不一定。

組員貢獻:  
Slave, Master: B06902065,B05902132,B06902042

Bonus: B06902117,B06902051

Report: B06902065,B05902132,B06902042

Bonus\_Report: B06902117,B06902051