Ch1

1 a. To initiate a DMA transfer, the CPU first sets up the DMA registers, which contain a pointer

to the source of a transfer, a pointer to the destination of the transfer, and a counter of the number of

bytes to be transferred. Then the DMA controller proceeds to place addresses on the bus to perform

transfers, while the CPU is available to accomplish other work.

2 b. Once the entire transfer is finished, the DMA controller interrupts the CPU.

3 c. Both the CPU and the DMA controller are bus masters. A problem would be created if both the

CPU and the DMA controller want to access the memory at the same time. Accordingly, the CPU

should be momentarily prevented from accessing main memory when the DMA controller seizes the

memory bus. However, if the CPU is still allowed to access data in its primary and secondary caches,

a coherency issue may be created if both the CPU and the DMA controller update the same memory

locations.

Ch2

4.1 shared memory model

message passing model

4.2 shared memory model: 行程使用 share memory create和 shared memory attach系統呼叫來產生以及取得其他行程所擁有的記憶體區域的存取權。因此要預防共用所會產生的問題，例如用critical section來解決，而確保不會對同一個區域進行寫入的動作。

優點：可取得最大的通訊速度及便利性，因為它是用記憶體轉換運作的速度來操作。

缺點：在行程分享記憶體的保護及同步方面仍存在許多問題。

Message passing model: 資訊藉由作業系統所提供行程間的聯繫來做交換。

優點：訊息傳遞方式只有在少數資料需要交換時比較有用，而且對電腦之間的通信而言，也比共用記憶體方式容易製作。

缺點：訊息傳遞模式每次傳遞訊息都需要經過作業系統，因此比較沒有效率，存取速度慢。

5.1 Easier to port the OS to new architectures

5-2 Communication takes place between user modules using message passing

5-3 Performance overhead of user space to kernel space communication

Ch3

6 Short-term (CPU scheduler)—selects from jobs in memory those

jobs that are ready to execute and allocates the CPU to them.

短排程(cpu排班程式)：從記憶體中的工作選擇出已準備好要執行，並配置cpu資源給那些工作。

Medium-term—used especially with time-sharing systems as an

intermediate scheduling level. A swapping scheme is implemented

to remove partially run programs from memory and reinstate them

later to continue where they left off.

中排程：特別用於分時系統中的排程階級。置換機制被建置用來移除部份在記憶體執行中的程式及恢復那些被暫時移除的程式。

Long-term (job scheduler)—determines which jobs are brought into memory for processing.

長排程(工作排班程式)：決定哪個工作被載入記憶體執行。

The primary difference is in the frequency of their execution. The shortterm

must select a new process quite often. Long-term is used much less

often since it handles placing jobs in the system and may wait a while

for a job to finish before it admits another one.

最主要的不同點就是它們執行的頻率。短排程必需經常選擇一個新的程序。長排程則只必須操控系統中工作的載入，然後可能等到另一個工作完成前再允許另一個工作進入。

(3.11)

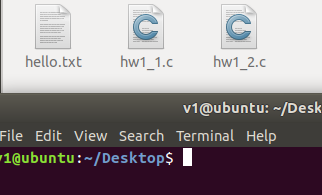
a A benefit of synchronous communication is that it allows a rendezvous between the sender and receiver. A disadvantage of a blocking send is that a rendezvous may not be required and the message could be delivered asynchronously. As a result, message-passing systems often provide both forms of synchronization.

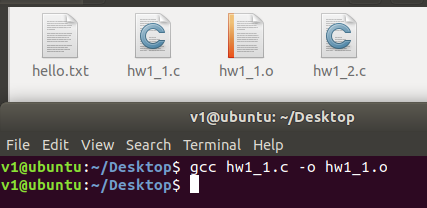
b Automatic buffering provides a queue with indefinite length, thus ensuring the sender will never have to block while waiting to copy a message. There are no specifications on how automatic buffering will be provided; one scheme may reserve sufficiently large memory where much of the memory is wasted. Explicit buffering specifies how large the buffer is. In this situation, the sender may be blocked while waiting for available space in the queue. However, it is less likely that memory will be wasted with explicit buffering.

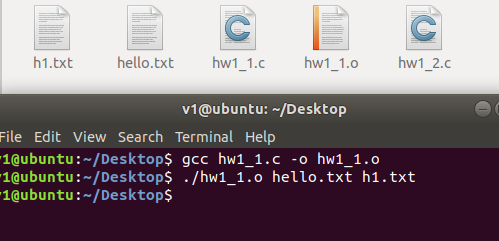
c Send by copy does not allow the receiver to alter the state of the parameter; send by reference does allow it. A benefit of send by reference is that it allows the programmer to write a distributed version of a centralized application. Java’s RMI provides both; however, passing a parameter by reference requires declaring the parameter as a remote object as well

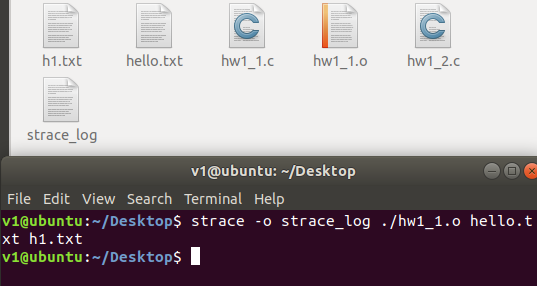
d The implications of this are mostly related to buffering issues; with fixed-size messages, a buffer with a specific size can hold a known number of messages. The number of variable-sized messages that can be held by such a buffer is unknown. Consider how Windows 2000 handles this situation: with fixed-sized messages (anything < 256 bytes), the messages are copied from the address space of the sender to the address space of the receiving process. Larger messages (i.e. variable-sized messages) use shared memory to pass the message.

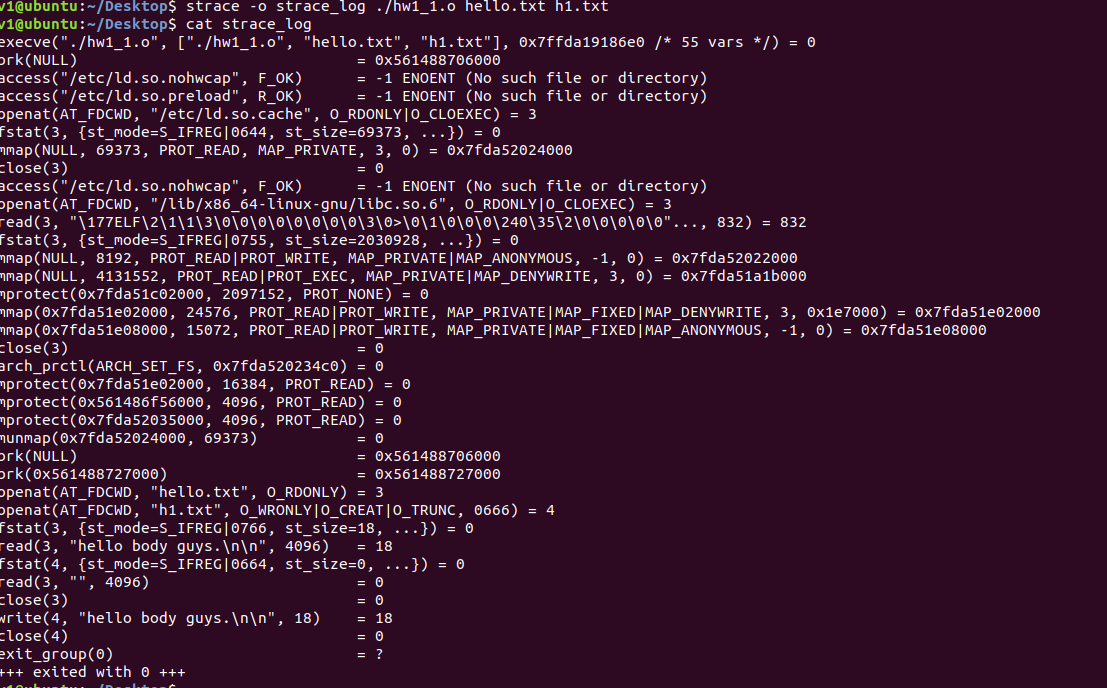
Program1

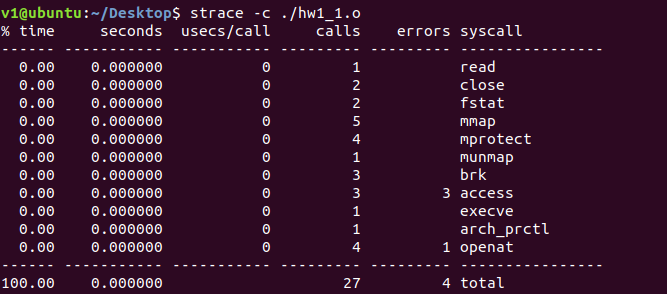




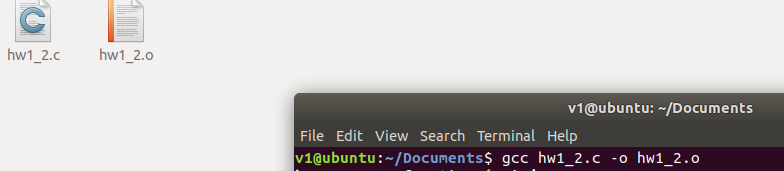


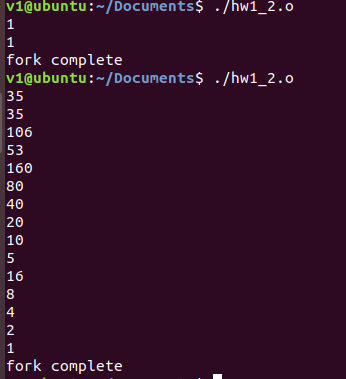






Program2

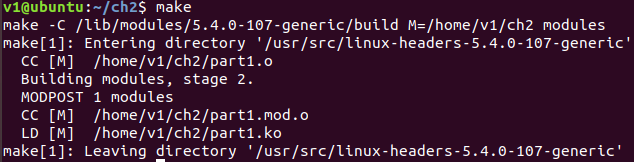


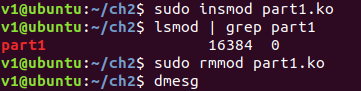


Program Project

Ch2

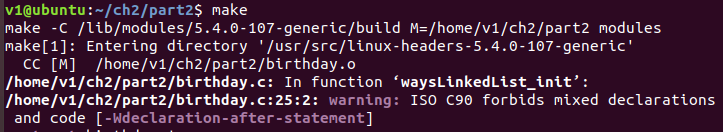
Part1

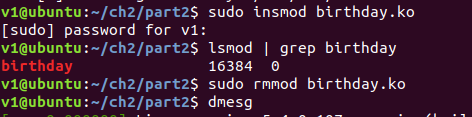


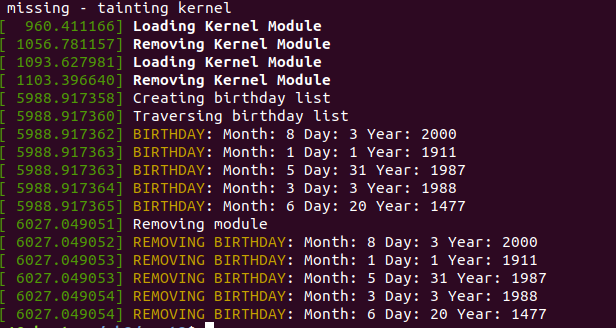




Part2







CH3