1.

(a) 可以利用GPGPU，將可平行化部份交由GPU來做，可有較好效能

(b) Embedded，裝在車上使用

(c) 8 teraflop of calculation power / 250W $\Rightarrow \frac{8}{250}$ *

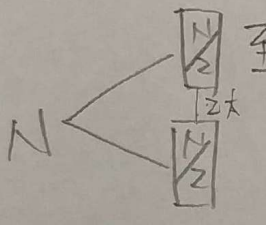
(d) 一般desktop computer雖有low response time但並不需要"即時、立刻"
裝在車上顧及安全性計算之PX-2屬於hard-real-time system，且desktop太耗能，其他設備的相容性也差.

(e) 利用多個node平行計算達加速效果

(f) 把每個layer當成一個stage，算完才執行下一層計算

(g) throughput

(h) 好處：每種操作更快，因高集成ISA或硬体上之ROM3.
壞處：不具有靈活度自修改碼，系統不易做調控(deep learning結果也不能即時反應).

(i) 
至多產出$(\frac{N}{2})^2$種結果，交換2部對方計算結果
$2 \cdot (\frac{N}{2})^2 = \frac{N^2}{2}$ *

(j) 計算花A ns.，有Q個processor，上傳/下載皆10GB.

(k) 在一台Machine上耗 $\frac{NA}{Q}$ ns
在二台Machine上耗 $\frac{NA}{2Q}$ ns) speed up = 2
在P台Machine上耗 $\frac{NA}{PQ}$) speed up = P

(l) 越緊鎖計算越少層
small = GPU、FPGA
medium = CPU, cluster (有少量也含有量)
large = custom-design chip (少量node)

2.

(a) No. 若Data存在其他cache可直接透过share bus共享，write要同時處理MEM<->cache和other

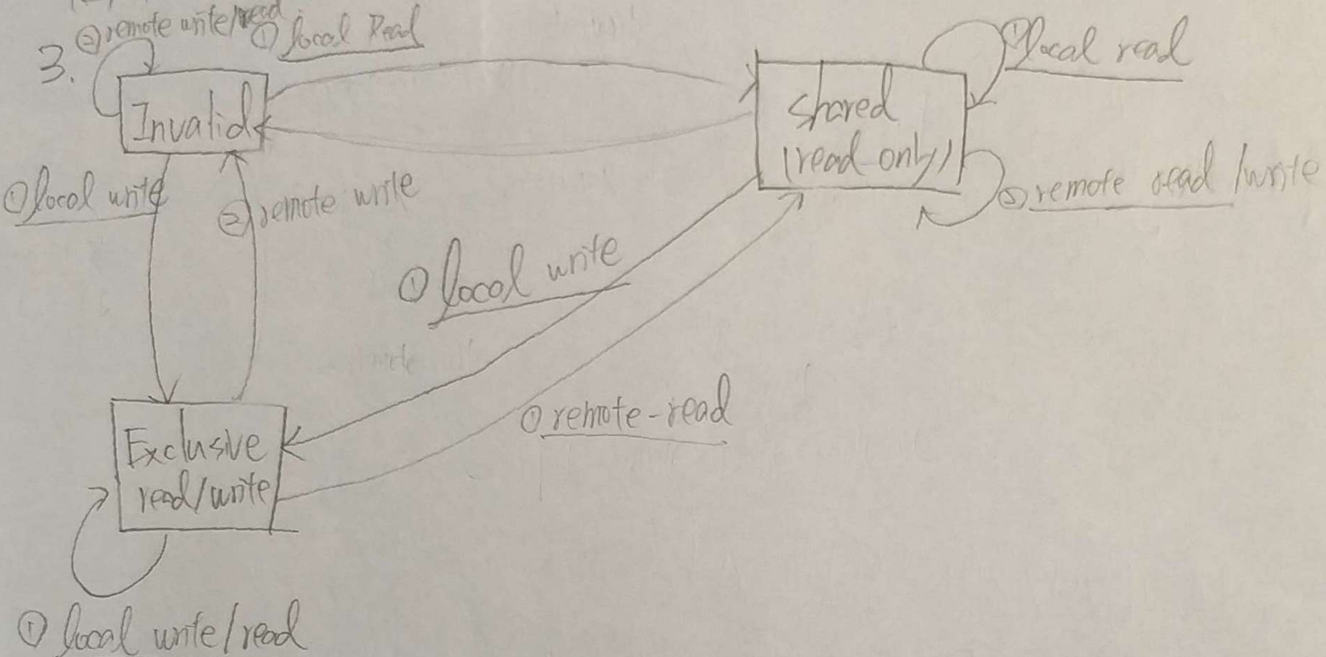
(b) Yes，兩處理器即便是存不相干变数仍有可能造成資料変动影響其他值

(c) NO，ABI依賴於硬体，較為低階，ABI用於制定硬体規範.

(D) Yes，会用到C語言之功能，故架構在C之上

(E) Yes，ARM的机器語言可用JVM的bytecode直表明，不需額外操作?

3.



用cache padding技術：讓每個会被多個cpu共用的变数獨佔一個cache line

4.

区块鏈金速，以太幣，去中心化加密货幣，平台.

(a) 是，適合GPU.

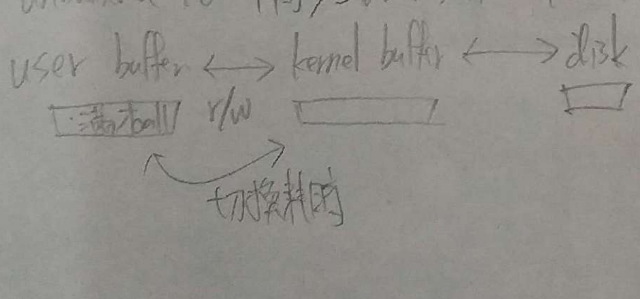
(b) 是，因data有相依無法利用GPU平行化效果

5. 因多個user同時对server進行存取，因為通道不同的延遲造成conflict.

6.

Buffered I/O：透过read()/write()實现，有多層架構調用，效率較適合cache等系統

unbuffered I/O：中間少了user buffer，kernel buffer仍存在，多次的write/read kernel buffer 才做disk讀寫



6.

(b) random access

(c) blocking I/O = request 到 kernel 就沒 return，中間經 wait for data / copy data

nonblocking I/O = request r/w 沒就 return，沒完成也會 return => 再 request

=> 要 nonblocking I/O

7. 事前建立好一些 thread 等待使用，若 thread pool 內無 thread，則 request wait，可重新使用

利用 working set window 輔助使資源使用率不低落.

8.

```
typedef struct{
    semaphore mutex = 1        // 互斥用
    semaphore hold_barrier = 0; // 卡至 n 個 thread 進入
    int counter = 0;
}barrier;
barrier_wait(barrier *b, int n)
```

基本互斥:

```
wait(mutex)
    counter++;
    if(counter == n) signal(hold_barrier).
signal(mutex)
```

條件達成，開始 work

```
wait(hold_barrier) // 卡到有 n 個 thread 到以放
```

基本互斥:

```
wait(mutex)
    counter--;
    if(counter >= 0) signal(hold_barrier)
signal(mutex)
```

放其他人直到沒 thread.

(b) 採用 priority scheduling + RR scheduling，不同 priority 有不用 time quantum，根據每一執行狀況動態調配 priority.

9. 對每個 user 和檔案建 access control matrix

	user 1	user 2
file1	1 1 1	0 1 1
file2	1 0 1	0 1 0
file3	1 0 0	0 0 0