Explain each stage of a 5 stage pipeline	Briefly explain what pipelining is
What is a control hazard?	What are two ways of preventing control hazards?
Briefly explain what branch prediction is	What is used to implement branch prediction and what does it do?
What is a data hazard?	$What \ is \ forwarding?$

Where we get all the components of the CPU working at the same time, with buffers that are flushed every clock cycle inbetween each stage, so that we can overlap the execution of instructions to increase overall clock speed.

IF - Fetch instruction from memory ID - Decode instruction; select registers EX - Perform an operation or calculate an address MEM - Access an operand in memory WB - Write to registers

1 If we have a branch at the ID stage, then the fetched

Pipeline bubbling and branch prediction instruction at the IF stage will have to be ignored all the way down the pipeline, wasting one full cycle and causing a bubble.

4 3

If we can remember what address a branch directed us to fetch A branch target buffer which maps the virtual address of one next from what it did when we executed that branch branch instruction onto the virtual address of the instruction that is branched to

previously, then we can pre-emptively load that instruction in the IF stage instead of fetching the instruction at the PC.

5

Where we add extra paths to the architecture to pass updated This is where we execute instructions in parallel that depend register values back to previous stages of the pipeline on each other

How can we exploit instruction level parallelism?	What does VLIW stand for?
How can we implement an out of order processor?	What does a fully associative cache store?
How does the CPU locate an item in a fully associative cache?	$What \ is \ temporal \ locality?$
What is spatial locality?	What are the three common cache replacement algorithms?

Very Long Instruction Word

Fetch multiple instructions per cycle
Have multiple ALU's to execute instructions in parallel
Have common registers and caches, since the instructions are
operating on the same data

10

9

Addresses and their corresponding data

Have a buffer that instructions are fetched into A scheduler to choose which instructions to execute at what times

A cache to store memory and register accesses until all instructions have finished so that the application can execute normally as though instructions were executed in parallel

12

11

The principle that if you use an address once, you may use it again soon e.g. loops

 $Hardware\ compares\ the\ input\ address \ with\ all\ stored\ addresses \\ (in\ parallel)$

If we get a match we have a hit
If no match we must go to main memory

14

13

Least Recently Used (LRU)
Round Robin
Random

The principle that if you use an address once, you are also likely to use addresses nearby e.e. arrays

Explain the write-through cache write strategy.	Explain the copy-back cache write strategy.
Why does a direct mapped cache usually use static RAM?	Briefly explain what a set associative cache consists of.
What is the advantage of using a set associative cache?	What two control bits are usually used in cache entries?
Explain what a compulsory cache miss is?	Explain what a capacity cache miss is?



Explain what a conflict cache miss is?	Name some stuff that the operating system must load and store on a context switch.
How is a multi-threaded processor most commonly presented to the operating system?	What are the three types of hardware multithreading?
Briefly describe coarse grain multithreading.	What extras does coarse grain multithreading require from the processor?
What can a context switch do to the cache?	Briefly describe fine grained multithreading.

- · Process ID
- · Program Counter
- · Stack Pointer
- \cdot General registers
- · Memory management information
- · Open file list (and positions)
- · Network connections

In a direct mapped or set associative cache, there is competition between memory locations for places in the cache. If the cache was fully associative, then misses due to this wouldnt occur.

26 25

- \cdot Coarse grain
- · Fine grain
- · Simultaneous MultiThreading (SMT)

As a processor with multiple cores (even though only one multithreaded core may be in the processor).

28 27

You don't need to change much in the processor, just make it abort instructions after a cache miss and have it store (and later load) the state of the thread.

The processor switches threads (a context switch) whenever an expensive operation is started (such as a memory load).

30 29

This involves interleaving the instructions of several threads.

When memory is accessed, instructions from other threads
will be executed to ensure stalls are brief. The aim is to reduce
the cost of switching CPU threads to almost nothing.

A context switch can trash the cache since the new thread may access different regions of memory and therefore all the memory reads will be misses. New values will be loaded into the cache which will destroy its previous data.

Briefly describe SMT.	What are the motivations that are driving us towards multi core systems?
What do different cores on a processor not share?	What is (my definition ^{TM} of) consistency?
What are the three special instructions used to guarantee out of order processors maintain consistency?	What is transactional memory?
What are the two most simple snooping protocols?	Describe 'Write update' (the snooping protocol).

	the same time. This requires significant hardware overhead, in balto yives you more freedom for instruction scheduling (since rinstructions in different threads are rarely interdependent so you can interleave them).	
34	33	
The programmer's view of the system. For example, they expect that if a memory location is updated in one thread, then the change will be visible across all threads, not just the threads that are running on the core that has the new value in its L1D cache.	An L1 cache (split into data and instruction caches) and sometimes an L2 cache. They also have their own registers obviously	
36	35	
Memory that supports transactions (yeah, duh). You can read and write to it however you like, but when you're finished, you have to commit, when your transaction is checked for conflicts and rolled back if it does conflict.	 A fence makes sure each memory access before the fence is A barrier makes threads wait until they have all reached the A lock makes sure that only one thread enters a critical sect 	e bar
38	37	
When a core writes a value to memory, the value is updated in its L1 cache, the cache then broadcasts the address on the bus,	$Write\ update\ and\ write\ invalidate.$	

and the snooping caches update their copy.

So many transistors per unit area, cooling is a massive issueWe have instructions from multiple threads in the pipeline at

Describe 'Write invalidate' (the snooping protocol).	Why is write invalidate better than write update for things like loops or writes to different words of the same cache line?
What does MESI stand for?	What is a directory based protocol with reference to multi core systems?
What are the concerns about a NoC (Network on a Chip)?	Buses are at any one time and are controlled by a that divides its use into fin one and in a future one.
Name five NoC architectures.	Describe the three types of NoC routing.

If a value is being frequently updated, then write invalidate needs to happen once, but write update needs to happen on every update, which wastes power and can saturate the bus. When a core writes a value to memory, the value is updated in its L1 cache, but sends a write invalidate message to the other caches which then invalidate the updated cache line in their copies.

42

A protocol where there is a directory that holds information on what each L1 cache holds. This lets cores talk on a P2P basis rather than all using one bus.

Modified, Exclusive, Shared, Invalid.

44 43

Buses are single usage at any one time and are controlled by a clock that divides its use into time slots. You can send in one slot and receive in a future one.

- \cdot Bandwidth
- · Latency
- $\cdot \quad \textit{Fault tolerance} \\$
- \cdot Area
- · Power dissipation

46

Minimal Always select the shortest path to-

 $wards\ the\ destination$

Oblivious Take a fixed path every time (really

imple!)

Adaptive Take the least congested route, com-

plex though and uses lots of power so

its rarely used.

 \cdot Crossbar

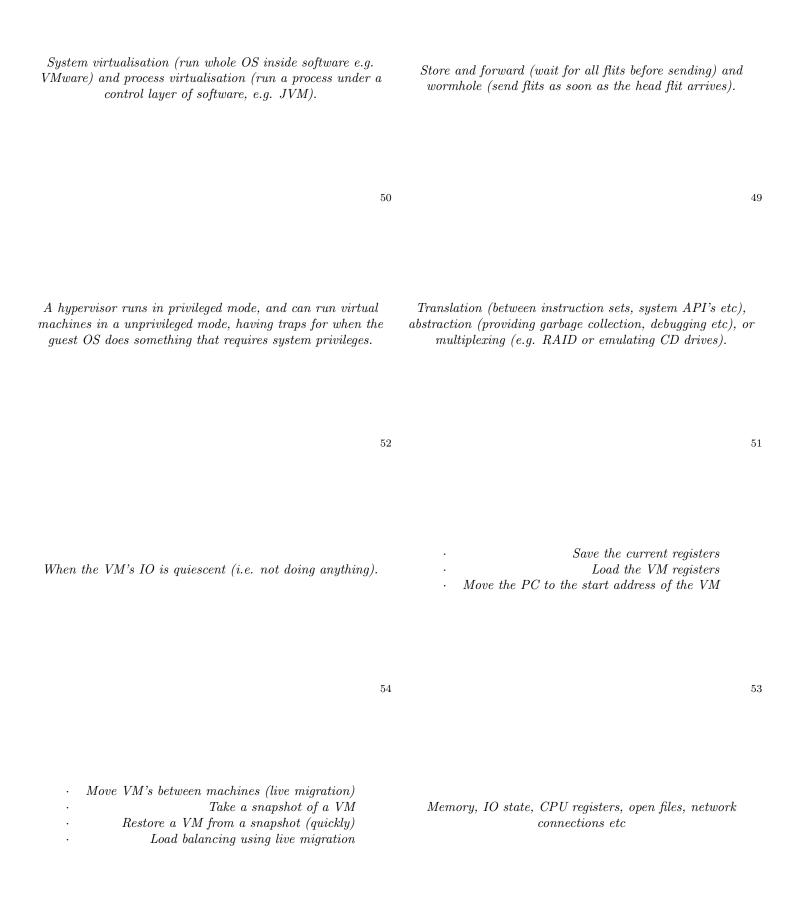
Ring

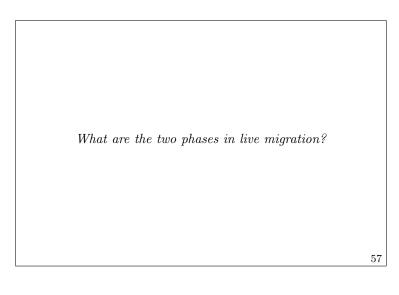
Tree

Fat Tree

Mesh

What are the two types of packet switching in NoC's?	What are the two types of virtualisation?
What are the three main advantages of virtualisation?	A hypervisor runs in mode, and can run virtual machines in a mode, having for when the guest OS does something that requires
What happens when you start a VM?	When is it best to stop a VM?
What is retained when a VM is stopped/paused?	What operations can we do on a VM?





The warm up phase and the stop and copy phase.