INDIAN INSTITUTE OF TECHNOLOGY, KHARAGPUR

End-Spring Semester 2018-19

High Performance Computer Architecture (CS60003)

Time= 3hrs **Marks=100**

Use of simple non-programmable calculators is permitted.

Special Instructions:

- **This question paper consists of two sections. You need to answer only one of the sections. All the two Year M.Tech students need to answer only Section A, and all other students need to answer only Section B.**
- **No clarification to any of the questions shall be provided. In case you have any queries, you can make suitable assumptions, but please write down your assumptions clearly.**
- All answers should be brief and concise. Lengthy and irrelevant answers will be penalized.

Section A

(Only two Year M.Tech students need to answer this section)

1. Write at least one advantage of a Physically Tagged, Virtually Indexed (PTVI) cache over the following two types of caches. Use one simple and short sentence for each.

a) A virtually tagged, virtually indexed (VTVI) cache **[1]**

b) A Physically tagged, physically indexed (PTPI) cache. **[1]**

2. Between invalidate and update snoopy cache coherency protocols, which one would provide shorter AMAT (Average Memory Access Time), other factors remaining the same? Justify your answer using one short sentence. **[1+1]**

3. Using at most one or two sentences explain why traditional semaphore primitives that are popularly being used in uniprocessors are considered unsatisfactory for use in multiprocessor systems. **[2]**

4. Consider a certain cache-coherent 4-core UMA multiprocessor, in which all processor cores have their private L1 and L2 caches, whereas the L3 cache is shared. In the design of L1 and L2 caches for the different cores, is it a good idea to implement "multi-level block inclusion" policy, or "multi-level block exclusion" policy? Justify your answer using one short sentence. **[2]**

5. Assume that a uniprocessor has an 8KB L1 cache and a 1MB L2 cache. Using one simple sentence, explain why the average latency experienced by the L2 cache controller on a main memory request may be much longer in case of a split transaction processor-memory bus as compared to what would be incurred for a traditional processor-memory bus. **[2]**

6. Consider a 64-processor CC-NUMA (Cache Coherent – Non Uniform Memory Access) computer system. Each processor has a 128KB cache. The sharing among the processors is at the block-level and the block size is 64B. The size of the main memory at each node is 2GB. A directory-based cache coherency protocol is being used.

a) What is the size of each directory entry (in bits)? **[1]**

b) What is the storage overhead (in percent) for the main memory for maintaining the directory entries? **[2]**

7. Consider a processor with a two-level cache. The L1 cache is direct-mapped and has bock size of 128KB. The L2 cache is 128KB, 8-way set-associative with block size of 128 bytes. To cut down the number of main memory (DRAM) accesses, the system designers are exploring the following two improvements to the L2 cache:

Improvement-1: Add one way to each set in the L2 cache, resulting in a 144KB 9-way set-associative L2 cache.

Improvement-2: Add a 128-entry fully-associative victim cache with the L2 cache.

a) What is the net addition to the cache capacity by each of the two improvements? **[1]**

b) Which one of the two improvements would be more effective in reducing the number of main memory (DRAM) accesses? Justify your answer using one or two sentences. **[3]**

8. Suppose you are running a program with two threads T0 and T1, which are running on different cores of a dual core processor. At a certain point during the run of the program, assume that both the threads T0 and T1 issue read commands for the variables **x**, **a**, and **b**. Immediately following this, the two threads execute one instruction each as shown below, and with side effects as listed below:

- **T0: x = 12;** (First, T0 writes x, which invalidates copy of that block in T1's processor cache)
- **T1: a = b;** (Next, T1 experiences a cache miss while reading the variable b)

As described above, a cache miss for T1 occurs on accessing the variable **b** due to the cache block invalidation caused due to the instruction executed by T0. Write the condition under which the cache miss for T1 would be considered as a true sharing miss. Under which condition would the cache miss be a false sharing miss? **[2+2]**

9. Copy the following table to your exam answer paper and fill in the blank cells in the table with the most appropriate statements from the list of statements given below, so that your table correctly summarizes the advantages and disadvantages of a given design change on the cache performance. Just filling in any one or more of a) to j) in the table cells as is appropriate, should be fine. **[4]**

a) Decreases capacity misses	f) Increases capacity misses
b) Decreases conflict misses	g) Increases conflict misses
c) Decreases compulsory misses	h) Increases compulsory misses
d) Decreases access time	i) Increases access time
e) Decreases miss penalty	j) Increases miss penalty

Design Change	**Potential advantage**	**Possible Disadvantage**
Increase cache size		
Increase block size		
Increase associativity		

10. A pipelined processor with 18 stages is to be used to run a program having 340 instructions. The processor uses a simple static "branch not taken" predictor. For the given program, branches comprise 20% of the instructions, and 40% of the branches are predicted correctly. Average penalty for each mispredicted branch is 1.7 cycles. Additionally, 2 percent of the total number instructions incur an average stall of 3 cycles on account of data hazard. Calculate the CPI for execution of the given program on this processor. **[5]**

11. Inspired by the success of the critical word first and early restart technique in reducing the L1 cache miss penalty, the designers for the Centurion processor are considering the use of this technique for the L2 cache. Assume that the Centurion processor has a 1 MB L2 cache with 64-byte blocks and a refill path that is 16 bytes wide. Further assume that the L2 cache can be written with 16 bytes every 4 processor cycles. The time to receive the first 16-bytes from the memory controller is 100 cycles, and transfer of each additional 16 bytes from the main memory requires 16 cycles. For each of the following two scenarios, determine how many cycles would it take to service an L2 cache miss?

a) Without using the critical word first and early restart technique? **[3]**

b) With the use of the critical word first and early restart technique? **[2]**

12. Assume that for a certain benchmark suite, the base CPI for a pipelined data path in a single core processor is found to be 1 with a perfect cache. Profiles of the benchmark suite for the *single core chip* with an L1 cache (code named C1) suggest that for every 10,000,000 accesses to the cache, there are 300,000 L1 cache misses. It was also determined that:

- If data is found in the L1 cache, it can be accessed in 1 clock cycle, and there are no pipe stalls

- If data is not found in the L1 cache, it can be accessed in 10 clock cycles on the average

Now, consider a dual-core version of the processor in which the private L1 cache at each processor is code named C2. All cores access a shared L2 cache. Coherency of L1 data is maintained by deploying an MSI coherency protocol. Profiles obtained by running the same benchmark suite on the dual-core system show that on an average, there are now 450,000 misses (considering both the L1 caches) for a total of 10,000,000 accesses.

- If data is found in a cache, it can still be accessed in 1 clock cycle
- However, on the average, 14 cycles are now required to satisfy an L1 cache miss.

What must the CPI of the multi-core system with a perfect cache, so that when fitted with C2 cache it will outperform the single core processor fitted with C1 cache? **[5]**

13. IIT(Kgp)'s ERP software runs on an 8-core multiprocessor system. The receivables of IIT(Kgp) consist of a large number of grants and also the student fees. In the ERP software, a thread is spawned whenever a receivable transaction is initiated. A thread corresponding to a receivable transaction updates a shared variable **balance**. For updating **balance** with the amount received, each thread essentially carries out the following operation:

balance = balance + amount; // You need to synchronize access to 'balance'!

Assume that the variable **balance** is shared among all the threads and its memory location is given by **0(R1)**. The received **amount** is local to the threads and is in the register **R2**. Write MIPS code using **LL** and **SC** instructions to achieve synchronized updating of the variable **balance** as per the balance updation operation given above. **[5]**

14. Assume that a uniprocessor has a single level 2-way set associative cache and incorporates way prediction. For a certain program, the cache exhibited 90% hit rate, and that the way-predictor was found to be right 75% of the time there is a cache hit. A cache hit with a correct way-prediction takes 2 cycles. On the other hand, a cache hit with an incorrect prediction takes 4 cycles, and a cache miss (way-prediction irrelevant) takes 60 cycles.

a) Compute the average memory access time of the cache with way-prediction. **[3]**

b) The original 2-way set associative cache (i.e. without way-prediction) has the same hit rate and miss time as the cache with way prediction, but has a 3 cycle hit time. By what factor does way-prediction improve the average memory access time? **[2]**

15. Consider a computer system with a two-level cache hierarchy. The L1 cache is split into instruction and data caches. The L2 cache is a unified cache. The cache parameters and statistics for a certain program being executed on this processor are as follows:

i. 40% instructions in the program are load/store instructions,

ii. L1 data cache hit ratio is 95%,

iii. L1 instruction cache hit ratio is 100%,

iv. L2 cache hit ratio is 70% (local),

v. Main memory access latency is 200 cycles,

vi. The program's CPI assuming a perfect L2 cache with no misses is 0.8,

vii. Each cache in the system is a blocking cache.

a) What is the L2 cache miss rate in terms of misses per 1000 instructions? **[3]**

b) What is the actual CPI for the given program? **[3]**

16. A processor has a single 4MB cache. For each of the following three optimization methods to be incorporated in the cache, identify how it would affect: a) miss rate, and b) memory traffic to the underlying memory. If it affects miss rate, identify the type of miss (out of 3C) it affects.

a) Nonblocking caches with "hit under miss" **[2]**

b) Hardware prefetching using stream buffer **[2]**
c) Way prediction **[2]**

17. Suppose you have written a program to run on a 32-core 2GHz shared-memory multiprocessor system.

a) The average CPI of a single core is 0.5, when all memory references are cache hits. On a simulated execution on a single core with a perfect cache, the program was observed to run for 30 minutes. How many instructions are executed during the run of the program? **[2]**

b) Suppose you want to reduce the run time of the program to 5 minutes with 8 core processors. Assuming that parallelization has no associated overhead, i.e. no extra instructions, no extra cache misses, no synchronization, communications, etc. What fraction of the program must be executed in parallel? **[3]**

c) Now consider that the main memory access time is 200 ns and a processor core stalls on a memory request till the request is serviced. Assume that 100% of the program can be executed in parallel and 1% of the instructions involve a cache miss, no matter how many processors are used. To make the program run in less than 10 minutes, at the least how many processor cores are to be configured for the run of the program? **[3]**

18. Consider a certain 24-node CC-NUMA computer, in which cache coherency is maintained at the block level through a directory-based MESI cache coherency protocol. Assume that all of the following operations are carried out by the processor n to a block whose home directory is processor p (n, p<24, and n≠p):

i. A read miss to an invalid line.
ii. A read miss to an exclusive line.
iii. A read miss to a shared line.
iv. A read miss to a modified line (processor m has the modified line).
v. A write miss to an invalid line.
vi. A write miss to a shared line.
vii. A write miss to a modified line (processor m has the modified line).
viii. A write hit to a shared line.
ix. A write hit to an exclusive line

a) Draw the states of the MESI protocol. For each of the above operations carried out by the processor (i.e. i to ix), indicate the state transition that takes place when the operation is carried out by drawing the transition between the appropriate states and annotating the operation number on it. For instance, if there is an operation in the above list with label x, and this operation causes the block to go from Shared to Invalid state, then in the figure you would add an edge from the Shared state to the Invalid state and you would label that edge with x. There might be multiple labels on a given edge in your figure, if more than one of the operations listed above cause the same transition. **[9]**

b) Which of the operation(s) listed above require(s) a multicast to different cache controllers in the CC-NUMA system? Explain for each of your identified operations, the cache controllers to which the message is sent. **[2]**

19. Suppose you are designing a 32-bit 5-stage pipelined processor. With a perfect cache, your simulation results show a CPI of 1. For selecting a cache for your processor, you are evaluating the performance of the following three cache systems, each with block size of 32 bytes. A "write buffer" is not used in any of the caches. You can assume that on the average 50% of the blocks are "dirty" and 20% of the instructions in the target program are "data transfer" instructions. The main memory (DRAM) has a latency is 40 cycles per memory request and only up to 4 bytes can be transferred per cycle on the processor-memory bus.

- **Cache 1:** A 16 KB direct mapped "unified" cache using "write-back". Miss-ratio = 2.9%. Does not affect the cycle length.

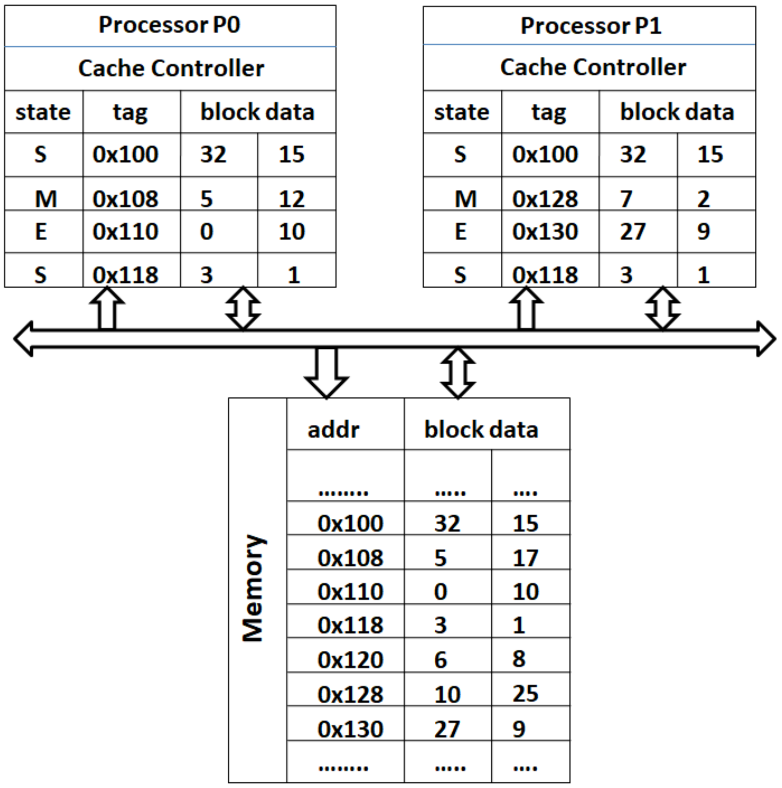
- **Cache 2:** A 16 KB 2-way set–associative "unified" cache using "write-back". Miss-ratio = 2.2%. Increases the cycle length with a factor 1.2
- **Cache 3:** A 32 KB direct mapped "unified" cache using "write-back". Miss-ratio = 2.0%. Increases the cycle length with a factor 1.25

a) Calculate effective CPI while using each of the three cache systems. **[3]**

b) Which of the above cache systems would you recommend from a performance perspective? **[3]**

c) What will be the AMAT for the cache 1 assuming that the TLB has a 0.2 % miss rate and a 20 cycle TLB miss penalty? The caches are physically addressed. **[3]**

20. The organization of a dual-core shared memory multiprocessor is shown below. Each processor has an L1 write-back private cache. Coherence is maintained using the MESI write-invalidate snooping protocol. For simplicity, each cache is directly-mapped with four blocks indexed from 0 to 3. Each block holds two words (8 bytes). For clarity, the tag contains the full address in hexadecimal, while the data is shown in decimal.



Processor P0 — Cache Controller

state	tag	block data	
S	0x100	32	15
M	0x108	5	12
E	0x110	0	10
S	0x118	3	1

Processor P1 — Cache Controller

state	tag	block data	
S	0x100	32	15
M	0x128	7	2
E	0x130	27	9
S	0x118	3	1

Memory

addr	block data	
……..	…..	….
0x100	32	15
0x108	5	17
0x110	0	10
0x118	3	1
0x120	6	8
0x128	10	25
0x130	27	9
……..	…..	….

The following memory operations are each executed at the initial cache and memory state given in the above figure. For each operation, please write bus transaction (if any) the resulting state, tag, and value of the caches and memory for every operation. Also for read transactions, write the value is returned by the read operation, and for the write operations, write the content of the relevant updated cache blocks.

a) P0 reads address 0x120 **[2]**

b) P1 reads address 0x114 **[2]**

c) P1 writes address 0x100 $\leftarrow$ 28 **[2]**

d) P1 writes address 0x130 $\leftarrow$ 16 **[2]**

e) P0 reads address 0x128 **[2]**

PLEASE SEE NEXT PAGE FOR PART B (ONLY TO BE ATTEMPTED BY DUAL DEGREE STUDENTS AND RESEARCH SCHOLARS)

Section B (Answer ALL the Questions)

(To be answered by Dual Degree MTechs and Research Scholars Only)

1. Consider the code snippet given below and answer the following questions.

```
I1: ADD R1,R2,R3
I2: SUB R2,R1,R2
I3: MUL R1,R3,R4
I4: MUL R4,R1,R3
I5: ADD R1,R3,R3
I6: DIV R5,R1,R3
I7: SUB R5,R6,R7
```

   (a) Identify all the true and false dependencies in the given code snippet. **[3]**

   (b) Remove all the false dependencies from the given code snippet by register renaming approach using Register Alias Table (RAT). Assume that the hardware has a pool of temporary registers (call them T registers, and assume there are 16 of them, T0-T15). **[4]**

   (c) Consider a 3-issue processor with out-of-order execution having one multiplication/division unit and two addition/subtraction units. Compute the values of Instruction Level Parallelism (ILP) and Instruction Per Cycle (IPC) for the given code snippet. **[2+3=5]**

   (d) Consider the same processor configuration as mentioned in the above question, but instead of out-of-order execution now assume the processor allows only in-order execution. Compute the Instruction Per Cycle (IPC) value with this configuration. **[3]**

2. (a) A five instruction sequence executes according to Tomasulo's algorithm. Each instruction is either of the two forms as mentioned below:

```
ADD <Dest Reg>,<Src Reg 1>,<Src Reg 2>
MUL <Dest Reg>,<Src Reg 1>,<Src Reg 2>
```

   `ADD`s are pipelined and take 2 cycles for execution i.e. E1-E2. `MUL`s are also pipelined and take 5 cycles for execution (E1-E5). There are only one `ADD` and one `MUL` unit present in the processor. An instruction must wait until its source registers are ready. For instance, if instruction 2 has a read-after-write dependence on instruction 1, instruction 2 can start executing in the next cycle after instruction 1 writes back. An example is shown below, where `F`, `D`, `WB` are fetch, decode and write-back respectively.

```
instruction 1 |F|D|E1|E2|WB|
instruction 2   |F|D |- |- |E1|.....|WB|
```

   The machine can fetch one instruction per cycle, and can decode one instruction per cycle. Consider the five-instruction program as follows.

```
ADD   R7,R6,R7
ADD   R3,R6,R7
MUL   R0,R3,R6
MUL   R2,R6,R6
ADD   R2,R0,R2
```

The initial contents of the register file are provided below.

R0	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**
4	5	6	7	8	9	10	11

i. In each cycle, a single instruction is fetched and a single instruction is decoded. There is no same-cycle `ISSUE`→`DISPATCH` or `CAPTURE`→`DISPATCH` operations. Assume the reservation stations are all initially empty, which are shown below. Put each instruction into the next available reservation station. For example, the first `ADD` goes into `RS0` and the first `MUL` goes into `RS3`. Instructions remain in the reservation stations until they are completed. Show the state of the reservation stations at each cycle till the end of cycle 10. [**Note:** When allocating source registers to reservation stations, please always have the higher numbered register be assigned to Source 2.] **[10]**

	ADD	
RS	**Source 1**	**Source 2**
RS0		
RS1		
RS2		

	MUL	
RS	**Source 1**	**Source 2**
RS3		
RS4		
RS5		

ii. Show the state of the Register Alias Table at the end of cycle 10. **[2]**

(b) Consider a program which consists of the following two instructions:

```
0x0040: LD   R0,[0xABCD]
0x0044: ADD  R1,R2,R3
```

Answer the following questions with the assumptions given below.

- An `LD` takes 3 cycles to execute and an `ADD` takes 1 cycle to execute.
- The `LD` and `ADD` can write their results to the Reorder Buffer (ROB) on the same cycle they are ready.
- It takes one cycle for the ROB to write data ready to be committed to architectural state to the architectural register file.
- The value at address `0xABCD` is 1.

i. Compute the contents of the architectural register file and the ROB for each of the next six cycles for a processor which can issue 1 instruction per cycle. The initial state of register file and ROB are given below. **[5]**

Register File			
R0	**R1**	**R2**	**R3**
5	4	3	2

Reorder Buffer		
Result Register	**Result Value**	**Done**
—	—	—
—	—	—

ii. *Assume* that on a system, without an ROB, an exception occurs when `ADD` is executed. What will be the content of architectural register file at the time of exception? Comment if there is any issue with this scenario. **[3]**

3. (a) Consider an example, where a `STORE` instruction writes the value `123` to memory address `0x1000`. A subsequent second `STORE` instruction copies the value that it finds at address `0x1000` (via a corresponding `LOAD`) to memory address `0x2000`. After executing both `STORE` instructions, both memory cells at the addresses `0x1000` and `0x2000` must contain the value `123`. The corresponding pseudo-code reads as follows:

```
STORE 123,0x1000
LOAD  register,[0x1000]
STORE register,0x2000
```

Answer the following questions.

i. What is the possible difficulty that the above memory operations may face due to their executions in a pipelined CPU? **[2]**

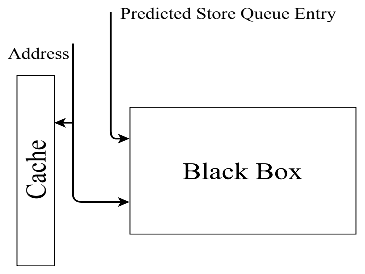
ii. Describe with proper explanation and diagram on how to remove the above difficulty using a store-to-load forwarding. **[4]**

iii. Assume a scenario when the initial `STORE` to `0x1000` gets delayed because of the dependency due to a prior instruction before the first `STORE`. This makes the store queue empty when the `LOAD` executes. What additional modification is required along with the store-to-load forwarding to tackle the situation? Explain with proper diagram. **[3]**

(b) In order to remove the associative search in the Store Queue as used in the store-to-load forwarding technique we predict a single in-flight store from which the load is most likely to forward. Assume that for a `LOAD` operation you are given the predicted Store Queue entry as another input along with the address. Design a circuit with proper explanation for the following black box module to remove the associative search in the Store Queue. The black box module will contain:

- One Decoder unit.
- One Comparator unit.
- One indexed Register File for implementing the Store Queue. The register file stores the tuple (address, data).

You will also need one Multiplexer unit along with the black box module to correctly get the data. **[6]**



4. The team to take the picture of the black-hole needs to process huge amount of information for which the following machine is being configured. It is a 64-bit byte-addressable system that supports virtual memory with the following detailed specifications:

- A page is 2K bytes.
- The page table is hierarchical with 3 levels.
- The first-level occupies 4 pages of memory.
- Each second-level occupies 8 pages of memory.
- A page table entry is 8 bytes and is the same for all levels.
- 32GB of physical memory is installed.

- Virtual addresses are 64 bits.

Answer the following questions with proper calculation steps.

(a) How many bits are used for the page offset? **[1]**

(b) How many virtual pages and physical frames exist in this system? **[2]**

(c) How many bits in a virtual address are used to index the first-level page table, second-level page table, and third-level page table? **[3]**

(d) To render the image of the black hole by an image processing program it requires processing 16TB of data. Compute the effective size of the 3-level page table required for this task. **[2]**

(e) Compute the effective size of a FLAT page table considering the task mentioned in question (d) and compare it with the 3-level page table. **[2]**

5. (a) A processor uses an 8K bytes 8-way set-associative L1 data cache with a 32-byte block size. This cache is virtually indexed and physically tagged (VIPT), and the processor's virtual address size is 64 bits but the physical address is only 48 bits. The cache keeps a tag for each line to compare it with the tag of the requested address. How many bits are needed in this entire cache to store all these tags? **[2]**

(b) Consider a system which has a 8-way set associative cache memory having 32 bytes of block size. The system has 4K bytes of page size. Assume that the cache is virtually indexed, physically tagged (VIPT) and has no aliasing problem. What is the maximum size of the cache? **[2]**

(c) Consider the table below for answering the following question. **[3+3=6]**

	Virtual Address Size	**Page Size**	**Page Table Entry Size**
i.	32 bits	4K bytes	4 bytes
ii.	64 bits	16K bytes	8 bytes

A cache designer wants to increase the size of a 4K bytes virtually indexed, physically tagged cache. Given the page size listed in the table above, is it possible to make a 16K bytes direct-mapped cache, assuming two words per block and each word is of four bytes? Answer for both the cases with proper calculation steps.

6. (a) Consider an 8-bit byte-addressable architecture that uses a 16-byte, 2-way set-associative cache with a block size of 4 bytes. The cache is initially empty, and uses LRU replacement policy. Fill out the following table for the given sequence of memory accesses, indicating the tag for each address as well as whether the access is a hit or a miss in the cache. If the access is a miss, indicate the type of miss (Compulsory / Capacity / Conflict) by placing a "✓" in the correct column. The first line is filled as an example. **[5]**

Address	**Hit/Miss**	**Compulsory**	**Capacity**	**Conflict**
0x13	Miss	✓		
0xAD				
0x02				
0x10				
0x85				
0x73				
0x01				

(b) A series of 12-bit address references to a cache of size 512 bytes are given in the following table. Fill in the following blanks to reverse-engineer the cache parameters. Provide a brief explanation for your answer. Assume that the block size and associativity of the cache are both powers of two, and it uses Least Recently Used (LRU) page replacement policy. **[5]**

Access	Memory Address	Cache Hit / Miss
1	0x720	Miss
2	0x920	Miss
3	0x90A	Miss
4	0xAB6	Miss
5	0x0A1	Miss
6	0x933	Hit
7	0xA3A	Miss
8	0x73F	Miss
9	0x92A	Hit

i. After access 3, it is clear that the block size must be less than ______ bytes.

ii. After access ______, it is clear that the block size is exactly ______ bytes.

iii. Given all accesses, it is clear that there are exactly ______ sets in the cache, and its associativity must be ______ -Way.

7. (a) Consider the following two cache designs:

   i. A 2-way set-associative cache.

   ii. A direct-mapped way-predicted cache with 85% prediction accuracy.

   For both the designs, cache hits take 1 cycle, cache misses take 20 cycles, and the cache miss rate is 0.005. A mispredicted way access that hits in the cache takes one more cycle. Compute and compare the average memory access time (AMAT) for both the designs. **[3]**

   (b) Consider a fully associative cache with 4 cache block and the following sequence of memory block requests. Answer the following questions.

   `2, 4, 1, 7, 2, 1, 3, 4, 7`

   i. Calculate the total number of cache hits if Least Recently Used (LRU) replacement policy is used. **[2]**

   ii. Calculate the minimum and maximum number of cache hits if Not Most Recently Used (NMRU) replacement policy is used. **[5]**

8. Consider a simple bus-based symmetric multiprocessor, where each processor has a single private cache with coherence maintained with a snooping, write-back protocol. Each cache is direct mapped, with 2 blocks each holding 1 word (e.g., addrs 0 and 2 are mapped to block B0, address 1 and 3 are mapped to block B1).

   For each of the following questions, the initial cache state is shown below. Show the resulting state of the caches and memory after each action. Show only the blocks that change. For instance, after Processor 0 reads 3, the changes are [P0.B1: (Shared, 3, 7)], indicating Processor 0s block B1 now has the state = Shared, address = 3, and data = 7. Also indicate the value returned by each read operation. **[10]**

Processor 0			
Block	**State**	**Address**	**Data**
B0	Modified	0	4
B1	Shared	1	5

Processor 1			
Block	**State**	**Address**	**Data**
B0	Modified	0	4
B1	Shared	1	5

Memory	
Addr	**Data**
0	9
1	5
2	8
3	7

(a) Processor 1 reads 2 (Processor 1 reads the content from address 2)

(b) Processor 1 writes 2 $\leftarrow$ 10 (Processor 1 writes 10 into address 2)

(c) Processor 0 writes 1 $\leftarrow$ 11 (Processor 0 writes 11 into address 1)

(d) Processor 1 writes 0 $\leftarrow$ 12 (Processor 1 writes 12 into address 0)

(e) Processor 0 reads 2 (Processor 0 reads the content from address 2)