Name:

Problem I (60 Points) : 
Problem II (50 Points) : 
Problem III (60 Points) : 
Problem IV (30 Points) : 
Problem V (60 Points) : 
Problem VI (50 Points) : 
Bonus Problem VII (40 Points) : 
Legibility and Name (5 Points) : 
Total (315 + 40 Points) : 

Instructions:

1. This is a closed book exam. You are allowed to have two letter-sized cheat sheets.
2. No electronic devices may be used.
3. This exam lasts 1 hour 50 minutes.
4. Clearly indicate your final answer.
5. Please show your work when needed.
6. Please write your initials on every odd page.
7. Please make sure that your answers to all questions (and all supporting work that is required) are contained in the space provided.

• Be cognizant of time. Do not spend too much time on one question.
• Be concise. You will be penalized for verbosity.
• Show work when needed. You will receive partial credit at our discretion.
• Write legibly. Show your final answer.
I. Potpourri (60 Points)

1) Dataflow and Vector Computers (5 Points)

Dataflow computers exploit ____________ parallelism; vector computers exploit ____________ parallelism.

2) Processing Paradigms (5 Points)

A/an ____________ processor executes the same ____________ with different data at the same time; a/an ____________ processor executes the same ____________ with different data at different times.

3) True Multiporting (5 Points)

In class, we discussed that adding a true second port to a memory cell increases access latency. Explain why this is so, concisely.

4) L1 Data Cache (5 Points)

A creative designer realizes one day that he can increase the L1 cache size of his design from 64KB to 2MB without affecting the number of cycles it takes to retrieve data from the cache and without reducing the processor frequency.

What can you definitively say about this design?
5) Sectored Cache vs. Smaller Blocks (10 Points)

After mounds of coffee and a good number of high-level simulation cycles, you narrowed down the design choices for the L1 cache of the next-generation processor you are architecting to the following two:

**Choice 1.** A sectored 64KB cache with 64-byte blocks and 8-byte subblocks

**Choice 2.** A non-sectored 64KB cache with 8-byte blocks

Assume the associativity of the two caches are the same.

What is one definitive advantage of the sectored cache over the non-sectored one?

What is one definitive advantage of the non-sectored cache over the sectored one?

Which of the choices has the faster cache hit latency? Circle one:

**Choice 1**  **Choice 2**  **Not Enough Information**

Justify your choice, describing the tradeoffs involved if necessary.
6) FastCaches, Inc. (10 Points)

Suppose a creative colleague of yours at FastCaches, Inc. proposed the following idea. Instead of directly using a number of bits to index into the cache, take those bits and form the index using a hash function implemented in hardware that randomizes the index.

What type of cache misses can this idea potentially reduce?

Explain briefly why.

What is a disadvantage of the idea other than the additional hardware cost of the hash function?
7) Dataflow versus Out-of-Order Execution (10 Points)

In class, we covered several dataflow machines that implemented dataflow execution at the ISA level. These machines included a structure/unit called the “matching store.” What is the function of the matching store (in less than 10 words)?

What structure accomplishes a similar function in an out-of-order processor?

When does the fetch of an instruction happen in a dataflow processor?

When does the fetch of an instruction happen in an out-of-order execution processor?
8) Page Table Bits (10 Points)

What is the purpose of the “reference” or “accessed” bit in a page table entry?

Describe what you would do if you did not have a reference bit in the PTE. Justify your reasoning and/or design choice.

What is the purpose of the dirty or modified bit in a page table entry?

Describe what you would do if you did not have a modified bit in the PTE. Justify your reasoning and/or design choice.
II. Vector Processing (50 Points)

You are studying a program that runs on a vector computer with the following latencies for various instructions:

- VLD and VST: 50 cycles for each vector element; fully interleaved and pipelined.
- VADD: 4 cycles for each vector element (fully pipelined).
- VMUL: 16 cycles for each vector element (fully pipelined).
- VDIV: 32 cycles for each vector element (fully pipelined).
- VRSHF (right shift): 1 cycle for each vector element (fully pipelined).

Assume that:

- The machine has an in-order pipeline.
- The machine supports chaining between vector functional units.
- In order to support 1-cycle memory access after the first element in a vector, the machine interleaves vector elements across memory banks. All vectors are stored in memory with the first element mapped to bank 0, the second element mapped to bank 1, etc.
- Each memory bank has an 8KB row buffer.
- Vector elements are 64 bits in size.
- Each memory bank has two ports (so that two loads/stores can be active simultaneously), and there are two load/store functional units available.

(a) What is the minimum power-of-two number of banks required in order for memory accesses to never stall? (Assume a vector stride of 1.)

(b) The machine (with as many banks as you found in part (a)) executes the following program (assume that the vector stride is set to 1):

\[
\begin{align*}
&\text{VLD } V1 \leftarrow A \\
&\text{VLD } V2 \leftarrow B \\
&\text{VADD } V3 \leftarrow V1, V2 \\
&\text{VMUL } V4 \leftarrow V3, V1 \\
&\text{VRSHF } V5 \leftarrow V4, 2
\end{align*}
\]

It takes 111 cycles to execute this program. What is the vector length?

If the machine did not support chaining (but could still pipeline independent operations), how many cycles would be required to execute the same program? Show your work.
(c) The architect of this machine decides that she needs to cut costs in the machine’s memory system. She reduces the number of banks by a factor of 2 from the number of banks you found in part (a) above. Because loads and stores might stall due to bank contention, an arbiter is added to each bank so that pending loads from the oldest instruction are serviced first. How many cycles does the program take to execute on the machine with this reduced-cost memory system (but with chaining)?

Now, the architect reduces cost further by reducing the number of memory banks (to a lower power of 2). The program executes in 279 cycles. How many banks are in the system?

(d) Another architect is now designing the second generation of this vector computer. He wants to build a multicore machine in which 4 vector processors share the same memory system. He scales up the number of banks by 4 in order to match the memory system bandwidth to the new demand. However, when he simulates this new machine design with a separate vector program running on every core, he finds that the average execution time is longer than if each individual program ran on the original single-core system with 1/4 the banks. Why could this be (in less than 20 words)? Provide concrete reason(s).

What change could this architect make to the system in order to alleviate this problem (in less than 20 words), while only changing the shared memory hierarchy?
III. DRAM Refresh (60 Points)

A memory system has four channels, and each channel has two ranks of DRAM chips. Each memory channel is controlled by a separate memory controller. Each rank of DRAM contains eight banks. A bank contains 32K rows. Each row in one bank is 8KB. The minimum retention time among all DRAM rows in the system is 64 ms. In order to ensure that no data is lost, every DRAM row is refreshed once per 64 ms. Every DRAM row refresh is initiated by a command from the memory controller which occupies the command bus on the associated memory channel for 5 ns and the associated bank for 40 ns. Let us consider a 1.024 second span of time.

We define utilization (of a resource such as a bus or a memory bank) as the fraction of total time for which a resource is occupied by a refresh command.

For each calculation in this section, you may leave your answer in simplified form in terms of powers of 2 and powers of 10.

(a) How many refreshes are performed by the memory controllers during the 1.024 second period in total across all four memory channels?

(b) What command bus utilization, across all memory channels, is directly caused by DRAM refreshes?

(c) What data bus utilization, across all memory channels, is directly caused by DRAM refreshes?

(d) What bank utilization (on average across all banks) is directly caused by DRAM refreshes?
(e) The system designer wishes to reduce the overhead of DRAM refreshes in order to improve system performance and reduce the energy spent in DRAM. A key observation is that not all rows in the DRAM chips need to be refreshed every 64 ms. In fact, rows need to be refreshed only at the following intervals in this particular system:

<table>
<thead>
<tr>
<th>Required Refresh Rate</th>
<th>Number of Rows</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 ms</td>
<td>$2^7$</td>
</tr>
<tr>
<td>128 ms</td>
<td>$2^9$</td>
</tr>
<tr>
<td>256 ms</td>
<td>all other rows</td>
</tr>
</tbody>
</table>

Given this distribution, if all rows are refreshed only as frequently as required to maintain their data, how many refreshes are performed by the memory controllers during the 1.024 second period in total across all four memory channels?

What command bus utilization (as a fraction of total time) is caused by DRAM refreshes in this case?

(f) What DRAM data bus utilization is caused by DRAM refreshes in this case?

(g) What bank utilization (on average across all banks) is caused by DRAM refreshes in this case?

(h) The system designer wants to achieve this reduction in refresh overhead by refreshing rows less frequently when they need less frequent refreshes. In order to implement this improvement, the system needs to track every row’s required refresh rate. What is the minimum number of bits of storage required to track this information?
Assume that the system designer implements an approximate mechanism to reduce refresh rate using Bloom filters, as we discussed in class. One Bloom filter is used to represent the set of all rows which require a 64 ms refresh rate, and another Bloom filter is used to track rows which require a 128 ms refresh rate. The system designer modifies the memory controller’s refresh logic so that on every potential refresh of a row (every 64 ms), it probes both Bloom filters. If either of the Bloom filter probes results in a “hit” for the row address, and if the row has not been refreshed in the most recent length of time for the refresh rate associated with that Bloom filter, then the row is refreshed. (If a row address hits in both Bloom filters, the more frequent refresh rate wins.) Any row that does not hit in either Bloom filter is refreshed at the default rate of once per 256 ms.

The false-positive rates for the two Bloom filters are as follows:

<table>
<thead>
<tr>
<th>Refresh Rate Bin</th>
<th>False Positive Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 ms</td>
<td>$2^{-20}$</td>
</tr>
<tr>
<td>128 ms</td>
<td>$2^{-8}$</td>
</tr>
</tbody>
</table>

The distribution of required row refresh rates specified in part (e) still applies.

How many refreshes are performed by the memory controllers during the 1.024 second period in total across all four memory channels?

What command bus utilization results from this refresh scheme?

What data bus utilization results from this refresh scheme?

What bank utilization (on average across all banks) results from this refresh scheme?
IV. Dataflow (30 Points)

What does the following dataflow program do? Specify clearly in less than 10 words (one could specify this function in three words).
V. Memory Request Scheduling (60 Points)

A machine has a DRAM main memory organized as 2 channels, 1 rank and 2 banks/channel. An open row policy is used, i.e., a row is retained in the row-buffer after an access until an access to another row is made. The following commands (defined as we discussed in class) can be issued to DRAM with the given latencies:

<table>
<thead>
<tr>
<th>Command</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVATE</td>
<td>15 ns</td>
</tr>
<tr>
<td>PRECHARGE</td>
<td>15 ns</td>
</tr>
<tr>
<td>READ/WRITE</td>
<td>15 ns</td>
</tr>
</tbody>
</table>

Assume the bus latency is 0 cycles.

(a) Two applications A and B are run on the machine. The following is a snapshot of the request buffers at time \( t_0 \). Requests are tagged with the index of the row they are destined to. Additionally, requests of applications A and B are indicated with different colors. Row 4 is initially open in bank 0 of channel 0 and row 10 is initially open in bank 1 of channel 1.

Each application is stalled until all of its memory requests are serviced and does not generate any more requests.

What is the stall time of application A using an FR-FCFS scheduling policy?

What is the stall time of application B using an FR-FCFS scheduling policy?
We studied Parallelism Aware Batch Scheduling (PAR-BS) in class. We will use a PAR-BS-like scheduler that we will call X.

This scheduler operates as follows:

- The scheduler forms request batches consisting of the 4 oldest requests from each application at each bank.
- At each bank, the scheduler ranks applications based on the number of requests they have outstanding at that bank. Applications with a smaller number of requests outstanding are assigned a higher rank.
- The scheduler always ranks the application with the oldest request higher in the event of a tie between applications.
- The scheduler prioritizes the requests of applications based on this ranking (higher ranked applications’ requests are prioritized over lower ranked applications’ requests).
- The scheduler repeats the above steps once all requests in a batch are serviced at all banks.

For the same request buffer state as in Part (a) (replicated above for your benefit):

What is the stall time of application A using this scheduler?

What is the stall time of application B using this scheduler?

The PAR-BS scheduler we studied in lecture provided better system performance than the FR-FCFS scheduler. Is this true for X also? (i.e., does X provide better system performance than FR-FCFS?)

Circle one: YES  NO
Explain why or why not. Provide the fundamental reason why X does or does not improve system performance over an FR-FCFS scheduler.

(c) Can you design a better memory scheduler (i.e., one that provides higher system performance) than X? YES  NO
If yes, answer the questions below.
What modifications would you make to scheduler X to design this better scheduler Y? Explain clearly.

What is the stall time of application A using this scheduler Y?

What is the stall time of application B using this scheduler Y?

(d) Consider a simple channel partitioning scheme where application A’s data is mapped to channel 0 and application B’s data is mapped to channel 1. When data is mapped to a different channel, only the channel number changes; the bank number does not change. For instance, requests of application A that were mapped to bank 1 of channel 1 would now be mapped to bank 1 of channel 0.
What is the stall time of application A using this channel partitioning mechanism and an FR-FCFS memory scheduler?

What is the stall time of application B using this channel partitioning mechanism and an FR-FCFS memory scheduler?
Explain why channel partitioning does better or worse than scheduler Y.

Is channel partitioning and FR-FCFS memory scheduling always strictly better or strictly worse than FR-FCFS memory scheduling alone? Explain.
VI. Caches and Virtual Memory (50 Points)

A four-way set-associative writeback cache has a $2^{11} \cdot 89$-bit tag store. The cache uses a custom replacement policy that requires 9 bits per set. The cache block size is 64 bytes. The cache is virtually-indexed and physically-tagged. Data from a given physical address can be present in up to eight different sets in the cache. The system uses hierarchical page tables with two levels. Each level of the page table contains 1024 entries. A page table may be larger or smaller than one page. The TLB contains 64 entries.

(a) How many bits of the virtual address are used to choose a set in the cache?

(b) What is the size of the cache data store?

(c) How many bits in the Physical Frame Number must overlap with the set index bits in the virtual address?

(d) On the following blank figure representing a virtual address, draw in bitfields and label bit positions for “cache block offset” and “set number.” Be complete, showing the beginning and ending bits of each field.

```
Virtual Address: ________________________________
```

(e) On the following blank figure representing a physical address, draw in bitfields and label bit positions for “physical frame number” and “page offset.” Be complete, showing the beginning and ending bits of each field.

```
Physical Address: ________________________________
```
(f) What is the page size?

(g) What is the size of the virtual address space?

(h) What is the size of the physical address space?
VII. BONUS: Memory Hierarchy (40 Points)

Assume you developed the next greatest memory technology, MagicRAM. A MagicRAM cell is non-volatile. The access latency of a MagicRAM cell is 2 times that of an SRAM cell but the same as that of a DRAM cell. The read/write energy of MagicRAM is similar to the read/write energy of DRAM. The cost of MagicRAM is similar to that of DRAM. MagicRAM has higher density than DRAM. MagicRAM has one shortcoming, however: a MagicRAM cell stops functioning after 2000 writes are performed to the cell.

(a) Is there an advantage of MagicRAM over DRAM other than its density? (Please do not repeat what is stated in the above paragraph.) Circle one: YES NO
Explain.

(b) Is there an advantage of MagicRAM over SRAM? Circle one: YES NO
Explain.

(c) Assume you have a system that has a 64KB L1 cache made of SRAM, a 12MB L2 cache made of SRAM, and 4GB main memory made of DRAM.
Assume you have complete design freedom and add structures to overcome the shortcoming of Magic-RAM. You will be able to propose a way to reduce/overcome the shortcoming of MagicRAM (note that you can design the hierarchy in any way you like, but cannot change MagicRAM itself).

Does it make sense to add MagicRAM anywhere in this memory hierarchy given that you can potentially reduce its shortcoming? Circle one: **YES**  **NO**

If so, where would you place MagicRAM? **Depict in the figure above clearly** and describe why you made this choice.

If not, why not? Explain below clearly and methodically.

(d) Propose a way to reduce/overcome the shortcoming of MagicRAM by modifying the given memory hierarchy. Be clear in your explanations and illustrate with drawings to aid understanding.

Explanation:

Figure(s):
SCRATCH PAD
SCRATCH PAD
SCRATCH PAD
SCRATCH PAD