ECE485/585 Homework No. 2 Solution

Problem No. 1

(a) Row decoder: Figures out the row number from the input address bits and then selects the desired row.
Wordlines: Activating a word line enables the pass transistors of all the SRAM cells in the row to be turned on, so that the contents of the row can be read/written.
Bitlines: Used to read/write the contents of the SRAM cells in the selected row
Column multiplexer: Used to select the data corresponding to the column number provided in the address bits

(b) Square SRAM arrays result in the wordlines and bitlines to have equal length. This results in minimum delay for the SRAM array.

(c) Due to DRAM's higher bit density, DRAM device capacities are much higher than SRAM capacities. Consequently, DRAMs need more address lines than SRAMs. Multiplexing the DRAM address lines (into row addresses and column addresses) reduces the pin count, which in turn reduces cost.

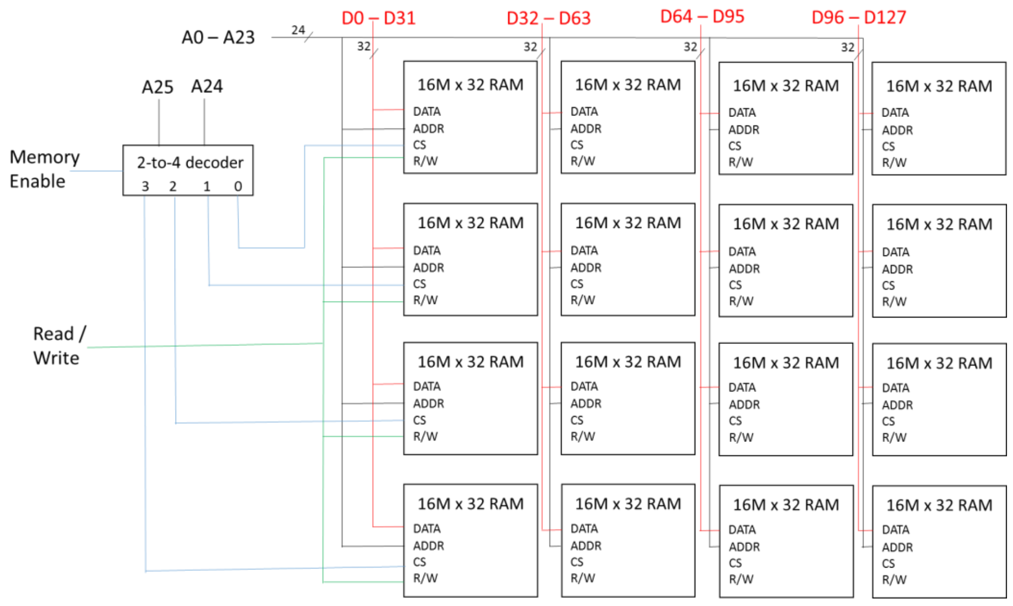
(d) DRAM reads are destructive. A read to a DRAM row causes all the DRAM cells in that row to transfer their charge to the bitlines and thereby lose their contents. Consequently, the cell contents need to be restored after every read operation.

Problem No. 2

(a) For each 16M x 32 memory chip: Chip capacity = 16M * 32 = $2^{24} * 2^5 = 2^{29}$ bits

For the entire 64M x 128 memory: Total memory capacity = 64M * 128 = $2^{26} * 2^7 = 2^{33}$ bits

Therefore, # of memory chips needed = Total memory size / Size of each chip = $2^{33}/2^{29} = 2^4 = 16$



(b)

Problem No. 3

(a) Channel width = 64 bits
Output width of each chip = 8 bits
Number of chips needed per rank = 64 / 8 = 8
Number of ranks = 4
Therefore, total number of DRAM chips needed = 4 * 8 = **32**

(b) Capacity of each chip = 8Gbit
Total number of DRAM chips = 32 (from part(a))
Therefore, total DRAM capacity = 8Gbit * 32 = 256Gbit = **32GBytes**

(c) Data transfer rate = Number of transfers per second * channel width * bus utilization
= $(1333*10^6)$ * (64/8 bytes) * (60%) = **6.4GB/s**

(d) Number of ranks = 4
Therefore, number of bits needed to address the **Rank** field = $\log_2(4)$ = **2**
Number of banks per rank = 16
Therefore, number of bits needed to address the **Bank** field = $\log_2(16)$ = **4**
Capacity of each row = 2Kbytes = 2^{11} bytes
Width of each column = Channel width = 64 bits = 8 bytes = 2^3 bytes
Number of columns per row = $2^{11} / 2^3 = 2^8 = 256$
Therefore, number of bits needed to address the **Column** field = $\log_2(256)$ = **8**

Capacity of each rank = 8Gbit * 8 = 64Gbit = 8GBytes = 2^{33} bytes
Number of banks per rank = 16 = 2^4
Therefore, capacity of each bank = $2^{33} / 2^4 = 2^{29}$ bytes
Capacity of each row = 2Kbytes = 2^{11} bytes
Therefore number of rows per bank = $2^{29} / 2^{11} = 2^{18}$
Hence, the number of bits needed to address the **Row** field = **18**

Problem No. 4

(a) Total number of DRAM rows in the system = 8GBytes / 4Kbytes = $2^{33} / 2^{12} = 2^{21}$
No. of refresh commands in a 64msec interval = 64msec / 7.8usec, which is roughly equal to 2^{13}
Therefore, in order to ensure that each row is refreshed once in every 64msec interval:
Number of rows refreshed in one refresh command = $2^{21} / 2^{13} = 2^8$ = **256**

(b) Since each refresh command triggers a parallel refresh operation in every bank, we need to calculate how many rows are refreshed in each bank during a single refresh command.
Number of banks = 16
Total number of rows refreshed in one refresh command = 256 (from part (a))
Number of rows refreshed in a single bank during one refresh command = 256 / 16 = 16
Latency of a single refresh command = 30nsec * 16 = **480nsec**

(c) t_{RFC} = 480nsec, t_{REFI} = 7.8usec
Fraction of time for which the memory system is unavailable = 480nsec / 7.8usec = **6.15%**

Problem No. 5

(a) Number of banks = 8
Therefore, number of bits needed to address the **Bank** field = $\log_2(8)$ = **3**
Capacity of each bank = 4Gbytes / 8 = $2^{32} / 2^3 = 2^{29}$ bytes
Capacity of each row = 4Kbytes = 2^{12} bytes
Therefore number of rows per bank = $2^{29} / 2^{12} = 2^{17}$
Hence, the number of bits needed to address the **Row** field = **17**
For every address, the most significant 17 bits provide the row number and the next 3 bits provide the bank number.

Address (hex)	Bank number	Row number (decimal)	Row buffer hit/miss	Comments
46A22000	2	36164	**Miss**	Row#36164 opened in bank#2
869D5440	5	68922	**Miss**	Row#68922 opened in bank#5
A43C2000	2	84088	**Miss**	Row#84088 opened in bank#2
59320680	0	45668	**Miss**	Row#45668 opened in bank#0
46A22040	2	36164	**Miss**	Row#36164 opened in bank#2
593204C0	0	45668	**Hit**	Hit in open row in bank#0
869D5000	5	68922	**Hit**	Hit in open row in bank#5

(b) Row buffer hit rate = Number of row bits / Number of accesses = 2 / 7 = **28.57%**

(c) DRAM clock speed = 667 MHz
Therefore 1 DRAM clock cycle = 1/667MHz = 1.5 nsec
A single burst requires 8 data transfers. There are 2 data transfers in one DRAM clock cycle. Therefore, t_{BURST} = 8/2 = 4 cycles
Row hit latency = $t_{CL} + t_{BURST}$ = 5 cycles + 4 cycles = 9 cycles
Row miss latency with no row currently open = $t_{RCD} + t_{CL} + t_{BURST}$ = 10 + 5 + 4 = 19 cycles
Row miss latency when there is an open row which first needs to be closed = $t_{RP} + t_{RCD} + t_{CL} + t_{BURST}$ = 10 + 10 + 5 + 4 = 29 cycles
Average DRAM latency = (9 * (2/7)) + (19 * (3/7)) + (29 * (2/7)) = 19 cycles = (19 * 1.5) nsec = **28.5nsec**

Note: The arrival time data for each request and the t_{RAS} values were supplied to you, so that you could account for queuing delays (if there were any). For the given arrival times, you can validate that whenever a request arrives at a bank, the previous request has already been completed (zero queueing delays). For example, the first request to bank#2 completes at 35ns, whereas the second request to the same bank arrives at 60ns. If the second request had arrived earlier, then it may have had to wait for the previous request to be completed, which could have resulted in additional latency.