Computer Architecture

Tutorial 2 – Main Memory Organisation

1 How many address bits are required for a 4G x 32-bit main memory if

(a) main memory is word-addressable?
(b) main memory is byte-addressable?

2 Suppose that a 1G x 32-bit main memory is built using 256M x 4-bit RAM chips and that this memory is **word-addressable**.

For this memory organisation evaluate:

a) the number of RAM chips per memory module?
b) the number of memory modules?
c) the number of RAM chips for the full memory?
d) the number of address bits needed for a memory module?
e) the number of address bits needed for the full memory?

In which memory module would memory word 14 (i.e. word address 14) be found when the memory system uses:

f) high-order interleave?
g) low-order interleave?

Assume memory modules are numbered from 0.

3 Suppose that the main memory given in question 2 is **byte-addressable**.

For this byte-addressable memory organisation evaluate:

a) the number of address bits needed for the full memory?

In which memory module would byte 14 (i.e. byte address 14) be found when the memory system uses:

b) high-order interleave?
c) low-order interleave?

Assume memory modules are numbered from 0.

4 Comment on the implications if we wished to transfer data between a little-endian memory (e.g. on an Intel computer) and a big-endian memory (e.g. on a PowerPC computer)?

Remember to show your working and carry out all the conversions without a calculator ☺

Computer Architecture

Tutorial 2 – Main Memory Organisation - Answers

1 (a) For a word addressable memory, we need to know how many rows (words) are there in memory.
Number of rows = 4G = 4 x 2^{30}= 2^2x 2^{30} = 2^{32}. To address 2^{32} rows, we need 32 bits.
Therefore **32 bits** are required to uniquely address each row and hence each 32-bit word.

(b) Each word is 32 bits = 4 bytes, so if main memory is byte-addressable we have,
Total number of bytes in memory = 4 x 4 Gigabytes i.e. 4 x 4 x 2^{30} = 2^2 x 2^2x 2^{30} = 2^{34}
Therefore **34 bits** are required to uniquely address each byte in memory.

2 a) RAM chips per module = Width of Memory/Width of each chip = 32bit/4bit = **8**
b) Memory Modules = Memory Rows in main memory / Memory Rows per Module = 1G/256M = **4**
c) Total number of Ram Chips = Number of Memory Modules x Number of Chips Per Module
= 4x8 = **32** or (1Gx32) / (256Mx4) = **32**
d) 256M module rows = 2^{28}. Therefore **28** address bits are needed (similar to question 1 (a)).
e) Memory is word-addressed, 1G words (rows) = 2^{30}. Therefore **30** address bits are needed (similar to question 1 (a)).
f) 14 **div** 256M (Module Length) = Memory Module **0**
g) 14 **mod** 4 (Memory Modules) = Memory Module **2**

3 a) If memory is byte-addressable, 1G rows x 4 bytes per row = Total number of bytes = 2^{30}x2^2=2^{32} .
Therefore **32** address bits are needed (similar reasoning to question 1 (a)).

For (b) & (c) we need *to divide the address by the number of bytes in a memory word* since we have byte addressing, e.g. in this case divide by 4 since there are 4 bytes in a 32-bit memory-word.

b) (14/4) **div** 256M (Module Length) = Memory Module **0**
c) Memory Module 2 **(see below)**

4 For some types & sizes of data we would have to re-order the transmitted data otherwise the data will not be what we sent. For example, a 16-bit Integer sent as 0x9E5A may be interpreted on the other machine as 0x5A9E, depending on the format. Normally, when sending data over a network, it is good practice to convert it from the sender machine's format into "network byte order" and then reconvert it on the other side back to the receiver machine's format. There are functions provided that do this transformation – learn more while doing network programming ☺.

Memory layout for question 3(c)

Module	Word				Byte			
0	0	0	0	0	0	4	8	12
	4	4	4	4	16	20	24	28
	8	8	8	8	32	36	40	44
	12	12	12	12	48	52	56	60
	…	…	…	…	…	…	…	…
1	1	1	1	1	1	5	9	13
	5	5	5	5	17	21	25	29
	9	9	9	9	33	37	41	45
	13	13	13	13	49	53	57	61
	…	…	…	…	…	…	…	…
2	2	2	2	2	2	6	10	14
	6	6	6	6	18	22	26	30
	10	10	10	10	34	38	42	46
	14	14	14	14	50	54	58	62
	…	…	…	…	…	…	…	…
3	3	3	3	3	3	7	14	15
	7	7	7	7	19	23	30	31
	11	11	11	11	35	48	46	47
	15	15	15	15	51	64	62	63
	…	…	…	…	…	…	…	…