Computer Science 605.411

Module 9 Example Set 4

1. Among the reasons that our MIPS system is referred to as a 32-bit system, is that it has 32-bit registers, 32-bit machine instructions and employs 32-bit addressess to access memory. The address bus and the data bus used to communicate between the CPU and the memory system are 32 bits wide as well. Each memory access transfers 32 bits of data from or into the memory. The lb (load byte) and the lh (load half-word) instructions are used to read one byte and two bytes respectively into some CPU register. Even when fewer than 4 bytes are requested (using lb or lh) a complete 32-bit word is read from memory and the appropriate bytes are extracted from the word and placed into the low byte or low two bytes of the register. The remaining high order bits in the register are filled with copies of the sign bit from the loaded byte or half-word.

a) If the memory system has a total size of 2 giga-bytes, how many 32-bit wide memory modules would be required if each module has a depth of 4194304?

A 2 GB memory contains a total of $2*2^{30} = 2^{31}$ bytes. Each module contains 4 bytes per cell and a total of 4194304 cells which corresponds to $4*4194304 = 2^{24} = 16777216$ bytes. Therefore the number of modules required = $2^{31}/2^{24} = 2^7 = 128$.

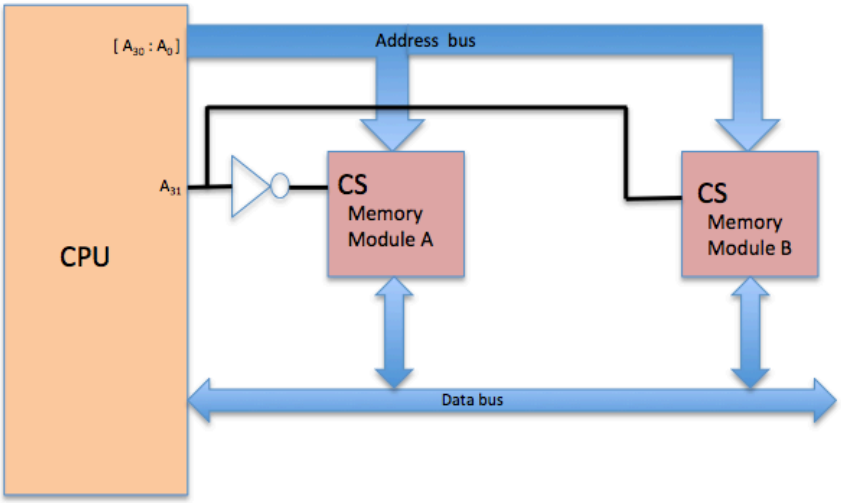
b) How would the bits within each 32-bit memory address be interpreted?

Since there are 2^{31} bytes of memory, the addresses range from 0 to $2^{31} – 1$ (i.e., 0 to 0x7FFFFFFF) and requires at most 31 bits. So the MSB within the 32-bit address will always be 0. The 32-bit memory address would be interpreted as follows:

0 MSB	7-bit module number	22-bit offset within module	2-bit offset within selected word

Only the 29 bits containing the 7-bit module number and 22-bit module offset need to be used to reference the selected memory word. The low 2 bits indicate the starting byte within the 4 byte word.

2. Assume that a very small memory is organized as shown in the following diagram:



Each memory module is 32 bits wide and has a depth of 32. The first byte within module A contains the value 0 and each subsequent byte contains a value that is 1 greater than the preceding byte. The first byte within module B contains the value 128 and each subsequent byte contains a value that is 1 greater than the preceding byte.

a) How many bytes are contained in each module?
Each cell is 4 bytes wide and there are 32 cells per module, so each module contains 4*32 = 128 bytes.

b) Assume that $t0 contains 0x80000046, $t1 contains 0x00000013 and $t2 contains 0x800000052. Describe what happens and what $t4 will contain when executing each of the following instructions:

I. lh $t4,0($t0)
The contents of the 2 bytes (i.e. half-word) starting at address 0x80000046 are read. Since the MSB A31 = 1, module B will be selected. The low 2 bits of the address = binary 10 and bits A30 through A1 = 00000000000000000000000010001. Hence cell 17 will be read and the high 2 bytes of the cell will be loaded into $t4. These are bytes 70 and 71 from module B which contain 128+70=198 and 128+71=199 respectively. Together they corrspond to the 16-bit pattern 0xC6C7 which is loaded into the low half of $t4 and sign extended, leaving 0xFFFFC6C7 as the final contents of $t4. Since the address 0x800000046 is even, no alignment exception will occur.

II. lw $t4,0($t2)
The contents of the 4-byte word starting at address 0x80000052 are read.
This address maps to module B and bits A30 through A1 = 00000000000000000000000010100, so cell 20 is read. The contents of cell 20 = 0xD3D2D1D0 and corresponds to bytes 80 through 83.
The low 2 bits = binary 10 indicate that the byte at offset 2 within cell 20 is the starting byte of the word to be read. However starting at offset 2, there are only 2 bytes remaining in the cell, hence an exception occurs to signal the error. The required word straddles two cells and can't be obtained in a single read operation.

III. lh $t4,0($t1)
The contents of the 2 bytes starting at address 0x00000013 are are the desired bytes.
This address maps to module A and bits A30 through A1 = 00000000000000000000000000100, so cell 4 is read. The contents of cell 4 = 0x13121110. The low 2 bits = binary 11 indicate that the byte at offset 3 is the starting byte. However the byte at offset 3 is the leftmost byte within the cell, so the required half-word can't be loaded into $t4 and an exception is triggerred because the two required bytes can't be obtain in a single read operation.

IV. lb $t4,1($t2)
The contents of the byte starting at address 0x80000052 is the desired bytes.
This address maps to module B and bits A30 through A1 = 00000000000000000000000010100, so cell 20 is read. The contents of cell 20 = 0xD3D2D1D0 and corresponds to bytes 80 through 83.
The low 2 bits = binary 10 indicate that the byte at offset 2 within cell 20 is the desired byte. Hence the value 0xD2 is loaded into the low byte of $t4 and sign extended through the remaining bits. So the final value in $t4 = 0xFFFFFFD2.