Computer Science 605.611

Problem Set 9

1. The total number of 16-bit memory cells contained in a small memory system = 1048576 (2^20). The memory contains 4 modules (modules 0 through 3). Each module is 16 bits wide. One way to view the memory is as a linear array of cells that are numbered 0 through 1048575.

a) (4) If the memory system employs low order interleaving, in which memory module will cell 35 reside and in which memory module will cell 2885 reside? Cells a numbered starting from 0.

Cell 35 resides in module \_\_\_\_\_3\_\_\_\_

Cell 2885 resides in module \_\_\_1\_\_\_\_

b) (4) If the memory system employs high order interleaving, in which memory module will cell 260000 reside and in which memory module will cell 732137 reside? Cells a numbered starting from 0.

Cell 260000 resides in module \_\_\_\_0\_\_

Cell 732137 resides in module \_\_\_\_2\_\_\_

2. Assume that register $t0 contains the value 0x400CD001. Indicate whether each of the following instructions does or does not attempt an unaligned memory access:

a) (2) lw $t1, 4($t0)

This does attempt an unaligned memory access. This is because it takes the value and more or less adds 4 to it (not actually what it does but is part of the instruction). Since the final address (0x400CD005) is not a multiple of 4 it would be unaligned.

b) (2) lb $t1, 7($t0)

This does not attempt an unaligned memory access. This is because load byte is not subjected to alignment restrictions.

c) (2) sh $t1, 5($t0)

This does not attempt an unaligned memory access. This is because the instruction stores a half word meaning the target address needs to be divisible by 2, the target address is 0x400CD006 which is a multiple of 2.

3. A certain computer has a memory system made up of four separate memory modules or chips. Each module has a width of 8 bits per cell and a depth of 16777216. The system contains a separate address bus and a separate data bus. The address bus and data bus are shared by the memory modules. On this system, memory accesses are performed in two phases.

The first phase is an addressing phase in which the shared address bus is used to select a module and to send the cell number to the selected memory module. The selected module saves or latches the cell number that appears on its address pins after which the address bus is released. While the specified cell within the selected module is being accessed, the address bus is free to be used for a transaction with a different memory module. Sixty nano-seconds are required to complete this first phase, the addressing phase.

Following the addressing phase for a selected module, a 60 nano-second data acquisition/transfer phase occurs in which the bits read from or to be written to the selected cell within the module are transferred over the shared 8-bit data bus.

One way to improve the performance of such a system is to “pipeline” the memory accesses by sending the address over the address bus to the next module to be accessed while the data from the currently accessed module is transferred over the data bus. While the data transfer occurs for one module, the address bus can be used to send an address to a different module. Recall that an individual module can only perform one read or write at a time and the address used by a module most not change while the read or write is in progress. Answer the following questions assuming that the address bus and data bus operations are overlapped in this fashion.

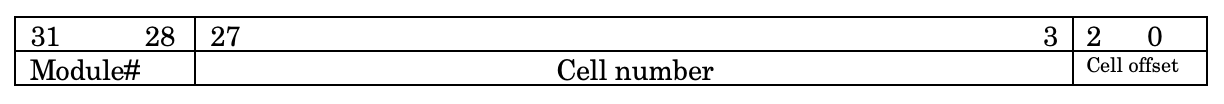
a) (3) What is the minimum number of nano-seconds required to read a 64-bit data item from this memory system if it employs high order interleaving?

The time required for this system employing high order interleaving to read a 64-bit data item is 480ns. This can be determined by the way the system is pipelined and handles the data flow. It first will be in the address stage which will take 60ns. Once that one moves into the data acquisition/transfer phase another 8 bit chunk will be added to the address phase meaning that it only requires 60ns for each subsequent instruction. So 60ns + (7\*60) = **480ns**

b) (3) What is the minimum number of nano-seconds required to write a 32-bit data item to this memory system if it employs low order interleaving?

The time that the system would take to write a 32 bit data item to this memory system if it employs low order interleaving is 300ns. Despite the difference in interleaving the pipelining of the system is more important. We can actually follow the same formula from the last equation, 60ns + (3 \* 60ns) = **300ns**

4. The diagram below shows the 32-bit address format used for a particular byte addressable memory system containing multiple memory modules:



a) (3) What is the width in bits of each memory module? Width = \_8 bits.

b) (2) To which memory module does the 32-bit decimal address 3405705229 correspond?

Modules are numbered starting from 0. Express your answer as a decimal integer.

Module Number = \_\_\_\_\_12\_\_\_\_\_\_\_

c) (3) What is the cell number to which address 0x00408000 corresponds?

Cells are numbered starting from 0. Express your answer as a decimal integer.

Cell Number = \_\_\_528384\_\_\_\_\_

d) (3) Does the memory system employ high order interleaving or does it employ low order interleaving?

This system employs high order interleaving as the higher bits of the address indicate the module number

5. Recall that the MIPS memory system is byte addressable and its CPU-to-memory bus is 32 bits wide. The memory cells are 32 bits wide and each module in the four-gigabyte (2^32 byte) memory contains 134217728 cells. Ignoring any exceptions that may be caused by the instructions, and assuming that high order interleaving is used, answer each of the following questions:

a) (3) If register $8 contains the bit pattern: 0xB003D8E0, how many memory cells must be transferred from the memory to the CPU to provide the data read by the following instruction?

lb $t0,45($8) Number of cells transferred = \_\_\_\_1\_\_\_\_\_\_

b) (3) If register $8 contains the bit pattern: 0xB003D8E0, how many memory cells must be transferred from the memory to the CPU to provide the data read by the following instruction?

lh $t0,45($8) Number of cells transferred = \_\_\_\_2\_\_\_\_\_

c) (3) If register $8 contains the bit pattern: 0xB003D8E0, how many memory cells must be transferred from the memory to the CPU to provide the data read by the following instruction?

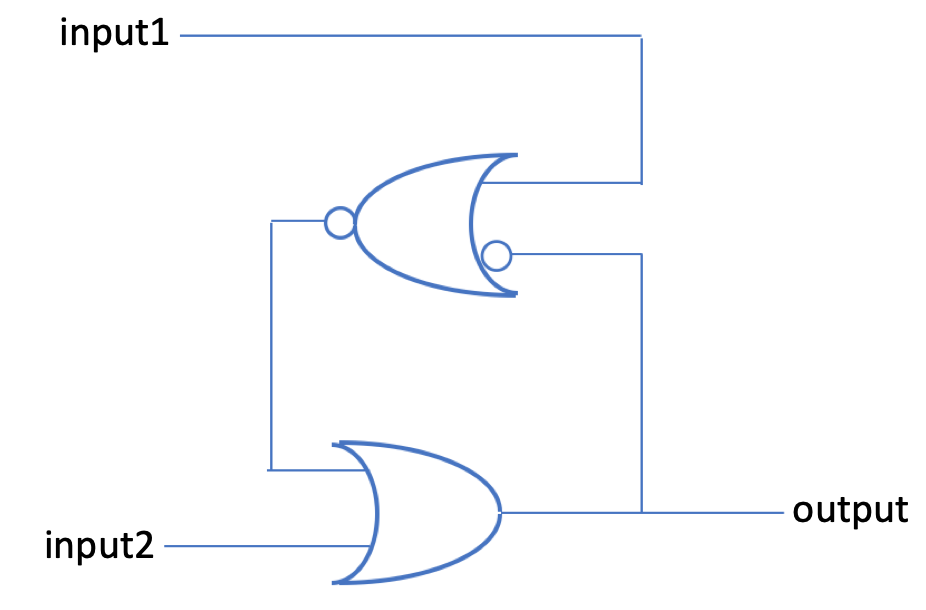
lw $t0,54($8) Number of cells transferred = \_\_\_\_1\_\_\_\_\_

d) (3) If register $8 contains the bit pattern: 0xC0000035, which module or modules must be accessed to obtain the data read by the following instruction?

lw $t0, -54($8)

This would access the fourth module.

6. Consider the circuit below which has a single-bit output and two single-bit control inputs: input1 and input2.



* 1. (3) Is this an example of a combinational circuit or is it an example of a sequential circuit? Explain your answer.

This is an example of a sequential circuit. This is because the output is fed back into the circuit. That means that it has a "memory” of the previous state indicating a sequential circuit.

* 1. (3) Assume that initially the output, input1 and input2 are all 0. If input1 alone is changed from 0 to 1, and then reset back to 0, can the final value of the output be determined? If so, does the final value for output = 0 or does the final value for output = 1?

The final value for output can be determined and is 0. This is because swapping just 1 back and forth will do nothing to the overall state of the output leaving it in state 0.

c. (3) Assume that initially the output = 1, and both input1 and input2 are 0. If input1 alone is changed from 0 to 1, and then reset back to 0, can the final value of the output be determined? If so, does the final value for output = 0 or does the final value for output = 1?

The final value for output would be 1 as the feedback would swap the input into the nor gate leaving the output in state 1.

d. (3) Assume that initially the output = 1, and both input1 and input2 are 0. If

input2 alone is changed from 0 to 1, and then reset back to 0, can the final value

of the output be determined? If so, is the final output 0 or is it 1?

The output should be 1. This is because swapping input 2 wont affect the value that is sent to the output as the output will continue to be 1 the entire time forcing the output to stay 1.

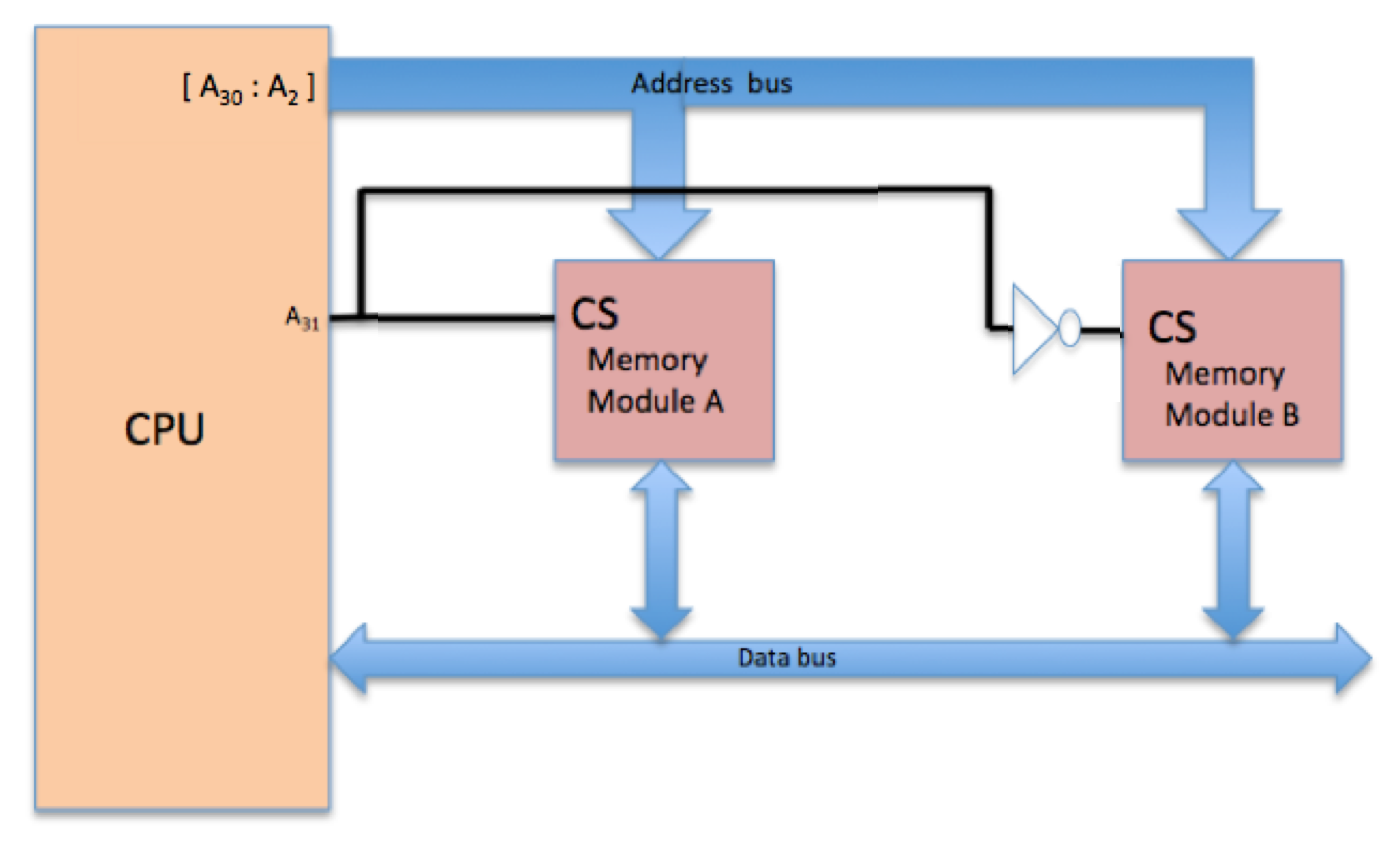
e. (3) Assume that initially the output = 0, and both input1 and input2 are 0. If

input2 alone is changed from 0 to 1, and then reset back to 0, can the final value

of the output be determined? If so, is the final output 0 or is it 1?

The final output would be 1. This is because swapping input 2 to 1 will set the output to 1 and then swapping it back will not affect the output.

7. Shown below is a block diagram (similar to the diagram in problem 5 on Module9\_ExampleSet3). It corresponds to a byte addressable memory system that contains two memory modules. Load and store instructions on this system employ 32-bit memory addresses to access memory. Both memory modules have a width of 32 bits and a depth of 536870912 cells. The cells within each memory module are numbered starting from 0. The bytes within each cell are numbered starting from 0 for the most significant byte (the leftmost byte) to 3 for the rightmost least significant byte. All control bits are active high (i.e., asserted when 1). Answer each of the following questions about this system:



a) (3) Notice that only the high order 30 address bits (bits A31 through A2) are used in the above diagram. What is the purpose of the two low address bits (A1& A0)?

The two low address bits allow for byte-addressable operations within the 32 bit wide cells of the memory modules. They are used to select the byte within the 32 bit word being accessed.

b) (3) Show the full 32-bit memory address that corresponds to byte number 52 within module B.

Byte 0 is the first byte within the module. Use eight hex digits to express your answer.

We can start by finding which word 52 is found in (52/4) which is 13. A31 has to be 1 so it ends up in module B. Finally we find the Byte number is 0 (52 mod 4). Combining these things we can find the address is 0x80000034.

c) (3) Suppose that a read from memory address 0x82345671 is performed. Use hex to show the bit pattern that appears on the memory address bus during the read operation.

d) (3) Use hex to show the 32-bit memory address that should be used by a lb (load byte) instruction to read the next to last byte within module B.

Address = 0x\_FFFFFFFE\_

e) (3) The following instruction sequence uses the lb (load byte) instruction to read the contents of a memory byte into register $4:

lui $4, 16384

lb $4, -3($4)

Which memory module (A or B) is accessed by this instruction sequence?

Module A

f) (3) The first cell in each memory module is assigned cell number 0. Indicate the cell number of the cell within the selected module that is accessed by the instruction sequence:

lui $4, 16384

lb $4, -3($4)

Cell # = \_268435455\_\_

g) (3) Assume that the 32-bit pattern 0xCAFEF00D is contained in the memory cell that is accessed by the instructions:

lui $4,16384

lb $4,-3($4)

Use hex to show the bit pattern that is placed on the data bus when these instructions are executed.

0xF0 is placed on the data bus.

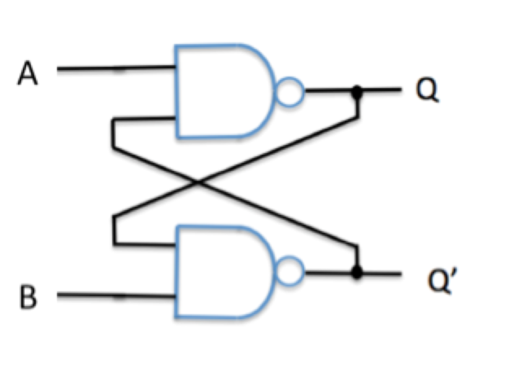
h) (3) Use hex to show the 32-bit memory address of the data byte that is placed into $4 by the instruction sequence:

lui $4, 16384

lb $4, -3($4)

Address = 0x3FFFFFFD\_\_\_

8. The diagram below shows a sequential circuit that has two control inputs A and B along with two outputs Q and Q’. The output from the upper NAND gate is defined as Q; the output from the lower NAND gate is defined as Q’.



a) (3) If Q = 1 and Q’=0, what happens to Q and Q’ if A and B are both 1?

Since Q=1 and Q’=0 and the inputs of A and B are both 1 we can determine what happens to Q and Q’. Q would remain 1 and Q’ would remain 0.

b) (3) If Q = 1 and Q’=0, what happens to Q and Q’ if A= 1 and B =0?

Both Q and Q’ will become 1 as each input will contain only one 1 and one 0 meaning they will always be 1 as the output.

9. The following sequence of instructions writes a 32-bit word to address 0x400C2000 within the MIPS memory:

lui $6, -20303 //Load -20303 into upper half of $6

ori $6, $6, -19789 //perform or between lower bits of $6 and -19789

lui $4, 16396 //load 16396 into upper half of $4

sw $6, 8192($4) //store $6 to address 0x400C2000

a) (3) Use hex to show the contents of the byte written at address 0x400C2001 if big-endian memory storage order is used.

If big endian storage order is used 0xB0B1B2B3 in order of address should look like this

0x400C2000 = B0

0x400C2001 = B1

0x400C2002 = B2

0x400C2003 = B3

**B1 should be contained in 0x400C2001**

b) (3) Use hex to show the contents of the byte written at address 0x400C2003 if little-endian memory storage order is used.

If little endian storage order is used it should look like this.

0x400C2000 = B3

0x400C2001 = B2

0x400C2002 = B1

0x400C2003 = B0

**So B0 should be contained in 0x400C2003.**

10. The four values 0xC0, 0xB6, 0x78 and 0x9A reside in memory at byte addresses: 0x400C2000, 0x400C2001, 0x400C2002 and 0x400C2003, respectively, and represent a 32-bit floating point number (0xC0B6789A). Hence the number is stored in big-endian order.

a) (3) Use hex to show the final 32-bit value in register $8 when the following instruction sequence is executed:

lui $8, 16396 / $8 = 400C0000

lhu $8, 8192($8) / $8 = 400C0000 + 8192 = 400C2000 = C0B6

srl $8, $8, 7 / $8 = 0x0181

andi $8, $8, 255 / $8 = 0x81

addiu $8, $8 ,-127 / $8 = 0xFFFE

**0xFFFFFFFE**

b) (3) Suppose that the same 32-bit floating point number had instead been stored in little-endian order starting at memory address 0x400C2000. Use hex to show the final 32-bit value in register $8 when the following instruction sequence is executed:

lui $8, 16396 / $8 = 0x400C000

lhu $8, 8192($8) / $8 = 789A

srl $8, $8, 7 / $8 = 0x00F1

andi $8, $8, 255 / $8 = 0xF1

addiu $8, $8, -127 / $8 = 0x72

**0x00000072**