Module 9 Example Set 1.

1. One meg equals what numeric value when used for:
a) memory storage capacity 1 MB = 2^{20} bytes (i.e. 1048576 bytes)
b) Clock rate 1 MHz = 1000000 cycles per second.

2. What determines how much data a memory system must supply in response to a read request?
The width of the data bus (the pathway between the CPU and the memory) determines the amount of data that the memory must supply in response to a read request. The memory returns 1 byte at a time for an 8-bit bus, two bytes at a time for a 16-bit bus, 4 bytes at a time for a 32-bit bus, etc.

3. What is meant by the "width" of a particular memory chip or module?
The width is defined as the number of data bits output from or taken into the chip for a single read or write operation. The number of data pins on the chip is equal to its width.

4. What is meant by the "depth" of a particular memory chip or module?
The depth is defined as the number of separate storage cells available within the chip. Each cell contains a number of bits equal to the width of the memory chip.

5. What is the total storage capacity of a memory chip that has a width of 16 and a depth of 1048576?
The total number of bits in the chip = 16 * 1048576 or 16 mega-bits. This is equivalent to (16 * 1048576)/8 = 2 mega-bytes.

6. How many address bits are required to reference a storage cell within a memory chip that is 32 bits wide and has a depth of 262144?
The number of address bits required to reference an individual 32-bit cell within the chip = $\log_2(262144) = 18$. Eighteen bits can cover the address range 0 to $2^{18} - 1$ (i.e. 0 to 262143).

7. On a byte-addressable memory system that employs 32-bit wide chips, how are the low order address bits used in reading a byte from memory?
Since the memory chips are 32 bits wide, 4 bytes are transferred for each access. Since each cell contains 4 bytes, the bits to the left of the two LSBs would be used to identify which cell to access within a chip. The two LSBs of the address would then be used to select the appropriate byte from the 4-byte group to send to the CPU.

8. A 32-bit CPU-to-memory data bus is used with a memory system made up of a collection of chips each of which has a width of 8 bits and a depth of 2^{24}. How many chips would have to be accessed in response to a single read request?
A group of 4 chips would be required to supply the 32-bit data (since 4*8 = 32).

9. What is the purpose of a chip select signal and how is it derived?
The chip select signal identifies which chips should respond to a requested memory operation. It is derived from one or more of the address bits used to the reference the memory system.

10. Suppose that our MIPS processor and a memory system communicate over a 16-bit bus. The memory system is made up of 512 memory chips each of which has a width of 1 and a depth of 2097152.

a) How many data transfers would be required in executing a lw instruction?
The lw instruction requires 32 bits of data from memory. Since the bus is only 16 bits wide, two separate 16-bit transfers are needed to obtain the 32 bits of data.

b) How many chips would have to respond to each read request?
Since each chip supplies only 1 bit, a group of 16 chips would each provide 1 of the bits for each 16-bit transfer.

c) How many of the address bits would be used in generating the chip select signal?
Since there are 16 chips in each group and there is a total of 512 chips, the number of groups = 512/16 = 32. So 5 of the address bits would be used to generate the chip select signal which would activate all 16 chips within a group.

d) How many of the address bits would be used to identify the storage cell within each chip? Since the depth of each chip is 2097152, the number of address bits needed to identify the cell to be accessed within each chip is $\log_2(2097152) = 21$. Note that the total storage capacity of the memory is 512 * 1 * 2097152 = 1073741824 bits or 64 MB. This requires 26 address bits, so only the low 26 bits of the 32-bit addresses generated by the processor will ever be non-zero. That is, the upper 6 bits of the 32-bit address register can be ignored or always set to zero.

11. If the CPU-to-memory bus is B bits wide and is driven by a clock with frequency F, what would be the bandwidth for the bus if one transfer per cycle is performed?
"Bandwidth" is defined as the amount of data transmitted per unit time. Hence the bandwidth = B*F, since F = 1/T where T is the clock period.

JOHNS HOPKINS UNIVERSITY | Engineering for Professionals

Computer Science 605.411

Module 9 Example Set 2

1. A memory system employs dynamic RAM that has a pre-charge time of 5ns and an access time of 25ns per read operation. Each read operation transfers 32 bits of data.

   a) What is the minimum cycle time for this memory system?

   The minimum cycle time = 30ns (5ns for pre-charging plus a 25ns access time).

   b) What is the bandwidth for this memory system?
   Each read operation transfers 4 bytes (32 bits) of data. Therefore, the bandwidth is 4 bytes/30ns = 1.333 x 10^8 bytes per second.

2. The null terminated character string "South Bridge" is stored in memory beginning at address 0x100C004. The first character in the string is "S", the second is "o", etc. The null character at the end contains eight 0 bits.

a) Assuming that big endian memory storage order is used, show (in hex) the contents of each of the following locations in this byte addressable memory:

Address	Contents
0x100C004	0x53="S"
0x100C005	0x6F="o"
0x100C006	0x75="u"
0x100C00B	0x72="r"

b) Assuming that little endian memory storage order is used, show (in hex) the contents of each of the following locations in this byte addressable memory:

Address	Contents
0x100C004	0x53="S"
0x100C007	0x74="t"
0x100C00A	0x42="B"
0x100C00F	0x65="e"

The string is stored from first character to last for both orders. The address of the string is the address of the leading (i.e., first) character in the string.

3. The contents of the memory bytes at locations 0x80000 through 0x80003 are as follows:

Address	Contents
0x80000	0xA9
0x80001	0x87
0x80002	0xD5
0x80003	0x43

Show, in hex, the contents of register $t0 produced by each of the instructions listed below assuming first that the byte addressable system employs big endian storage order and then that it employs little endian storage. Register $t1 contains the value 0x80000, **lw** is load word, **lh** is load half word, **lhu** is load half word unsigned, **lb** is load byte, and **lbu** is load byte unsigned.

	Big endian	Little endian
lw $t0,0($t1)	0xA987D543	0x43D587A9
lhu $t0,2($t1)	0x0000D543	0x000043D5
lb $t0,3($t1)	0x00000043	0x00000043
lbu $t0,0($t1)	0x000000A9	0x000000A9
lh $t0,0($t1)	0xFFFFA987	0xFFFF87A9
lh $t0,2($t1)	0xFFFFD543	0x000043D5

4. A memory system contains 16 modules that are organized so as to create a low order interleaved memory. Each memory module is 8 bits wides and the total storage capacity of 2 GB is provided collectively by the 16 modules. Recall that one GB is 2^{30} bytes.

a) How many bits would be required within the address for the module number?
Four bits would be required to specify module numbers 0 through 15. With low order interleaving, the module number resides in the low or rightmost bits.

b) What is the minimum number of bits required for the offset field within the address to allow any location within a memory module to be accessed?
Since there are 16 modules, the number of bytes within each module would be $(2*2^{30})/16 = 2^{27}$ hence 27 bits are required for the offset.

c) Write an instruction sequence that would fill each location within module 3 with all 1 bits.
The module number is 3, hence the address range for module 3 would have the binary pattern 0011 in the rightmost 4 bits and the 27 bits to the left of the module number field would range from 0 to 134217727. The following instruction sequence would work:

```
            li      $t2,255           # put all 1's into the low byte of $t2
            li      $t3, 134217727  # set count to fill all bytes within module
           li      $t4,3              # address of first byte in module 3
loop:       sb      $t2,0($t4)         # fill next byte in module with all 1's
            addiu  $t3,$t3,-1          # decrement loop counter
            bge     $t3,$0,loop        # repeat until all bytes have been written
            addiu  $t4,$t4,16          # increment offset within address
```

Note that the code takes advantage of the branch delay slot to increment the offset field by adding 16 (which has the effect of leaving the module number fixed and changing only the offset field).

5. a) An interleaved memory system contains four memory modules each of which is 32-bits wide and each module has a separate MDR (memory data register). The CPU communicates with the memory over a 32-bit bus. For each instruction fetch, the appropriate memory module takes 8 clock cycles to obtain the instruction and place it into the MDR associated with the memory module. One clock cycle is required to transfer the contents of the MDR to the CPU. What would be the minimum number of cycles required for the CPU to fetch a sequence of 8 consecutive instructions from this memory system?

Since the instructions are consecutive, they would appear in adjacent memory words each of which is in a different memory module. The interleaved memory can read 4 instructions into the 4 separate MDRs in parallel. Hence it would take 8 cycles to place the first 4 instructions into the respective MDRs. However, since the bus is only 32 bits wide, only one instruction at a time could be transferred from a MDR to the CPU. So it would take 8 cycles to read the first 4 instructions into the MDR then 4 cycles to transfer the instructions (one after the other) to the CPU. While the instructions are being transferred from the MDRs to the CPU, the read of the next 4 instructions could be started in parallel with these transfers. Thus the first 4 cycles of the next read would overlap with the 4 MDR transfers leaving only 4 cycles remaining until the next 4 instructions are placed into the MDRs. Therefore the minimum total number of cycles needed to fetch all 8 consecutive instructions would be 8 + 4 + 4 + 4 = 20 cycles (8 cycles to read the first 4 instructions, 4 cycles to transfer the MDR contents and start the second read, 4 cycles to complete the second read and 4 cycles to transfer the MDR contents to the CPU) .

b) A 32-bit wide memory module requires 8 clock cycles to read out an instruction into its MDR and 1 cycle to transfer the contents of the MDR to the CPU over a 32-bit bus. What is the minimum number of clock cycles required for the CPU to fetch a sequence of 8 consecutive instructions from this single memory module?

The single memory module can only read one instruction at a time. Hence the first instruction would require 9 cycles to reach the CPU (8 cycles to fill the MDR and 1 cycle to transfer the MDR contents to the CPU). While each instruction is on its way to the CPU, the read of the next instruction could be in progress. Hence each instruction after the first would arrive at the CPU 8 cycles following the preceding instruction. Therefore the minimum total number of cycles required to fetch all 8 consecutive instructions would be 9 + 7*8 = 65 cycles.

6. Thirty-two bit addresses are used to access a byte addressable data memory with a storage capacity of 256 MB. The memory is implemented using some number of memory modules each of which has a width of 32 and a depth of 1 M (i.e., 1048576).

a) How many memory modules are required for this data memory?

A cell within a module is 32 bits (i.e., 4 bytes) wide. Hence each module contains 4*1048576 bytes (4 MB). So a total of 256 MB / 4 MB = 64 modules are required.

b) Explain how a 32-bit address would be used to read a single byte from this data memory.

The low order 2 bits within the address would be used as the offset within the 32-bit memory cell selected by the remaining address bits. Each read would obtain the 4 bytes contained in the selected 32-bit cell and the low 2 bits would indicate which byte from the group of 4 to use.

c) What is the address of the next to last byte in module 33 (counting from 0)?

Since each module has a depth of 1048576, a 20-bit offset would be required to indicate which cell within module 33 to select. The low order 2 address bits indicate the byte offset within the selected cell. The module number would be indicated by the next 6 bits to the left of the cell offset field. Hence the address format would be:

Four 0 bits	6-bit module number	20-bit cell offset within module	2-bit byte offset within selected cell

Since the data memory is 256 MB in size, the corresponding address range is 0x00000000 to 0x0FFFFFFF. So the leftmost 4 address bits would always be 0.

The next to last byte within each 4-byte cell has offset 2. The final cell in each module would correspond to the maximum offset (0xFFFFF). So the next to last byte within a module would be byte 2 within the final cell which has the offset 0xFFFFF. Hence the address of the next to last byte in module 33=0x21 = binary 100001 is:

00001000011111111111111111111110 = 0x087FFFFE

Computer Science 605.411

Module 9 Example Set 3

1. A byte array that contains two 8-bit elements is declared as:
   unsigned char bdata[2];

   The element bdata[0] contains the value 0x2E
   The element bdata[1] contains the value 0x7F

   The beginning memory address for the array is 0x100900B0.
   Indicate in hex, the contents of the memory byte at location 0x100900B0 and at location 0x100900B1 for

   a) a little endian memory system:

Address	Contents
0x100900B0	0x2E
0x100900B1	0x7F

   b) a big endian memory system

Address	Contents
0x100900B0	0x2E
0x100900B1	0x7F

The elements of an array are stored adjacent to each other in memory beginning with the element with the lowest index followed by the element with the next higher index, etc. In this case each element is a single byte, so the memory endianness makes no difference. The endianness determines the order of the bytes within a multi-byte item in memory.

2. An integer array that contains two 32-bit elements is declared as:
   Unsigned integer idata[2];
   The element idata[0] contains the value 0x11223344
   The element idata[1] contains the value 0x05060708

   The beginning memory address for the array is 0x100900C0.
   Indicate in hex, the contents of the memory byte at location 0x100900C2 and at location 0x100900C5 for
   a) a little endian memory system

Address	Contents
0x100900C2	0x22
0x100900C5	0x07

The array elements are stored consecutively beginning with the first element. However, in this case each element contains 4 bytes. With little endian storage order, the low byte within each element is stored at the lowest address, followed by the next higher byte: 0x44, 0x33, 0x 22 and 0x11 for idata[0] and 0x08, 0x07, 0x06 and 0x05 for idata[1].

   b) a big endian memory system

Address	Contents
0x100900C2	0x33
0x100900C5	0x06

With big endian storage order, the high byte within each element is stored at the lowest address, followed by the next lower byte: 0x11, 0x22, 0x33 and 0x44 for idata[0] and 0x05, 0x06, 0x07 and 0x08 for idata[1].

3. Assume that our single-cycle datapath is required to support the MIPS instruction subset and to sustain a theoretical instruction execution rate of 200 MIPS. For the questions below, you may ignore all times except that required for the memory reads or writes.

a) What would be the maximum cycle time that the memory could have?
200 MIPS would correspond to one instruction every $1/2*10^8 = 2$ ns. The most time consuming instruction would be either a sw or lw because they are the only instructions in the subset that use a memory operand. So two memory operations (a fetch and an operand read or write) would have to be completed in a single clock cycle. Therefore the memory cycle time must be no more than 1ns.

b) What bandwidth, in bytes per second, is required for the CPU-memory bus to support this single-cycle system? That is, how many bytes per second must the bus carry?
Since two 32-bit (i.e., 4-byte) transfers must be performed in 1ns, the bandwidth = 4 billion bytes per second.

c) If the memory could only support a bandwidth of 128 million bytes per second over the 32-bit data bus, what would be the maximum theoretical MIPS rating for the single-cycle system?
Since the data memory transfers at most 4 bytes at a time, the bandwidth of 128 million bytes/sec would correspond to 32 million memory accesses per second. At two memory accesses per instruction, this would correspond to 16 MIPS.

4. The contents of the memory bytes at locations 0x80000 through 0x80003 are as follows:

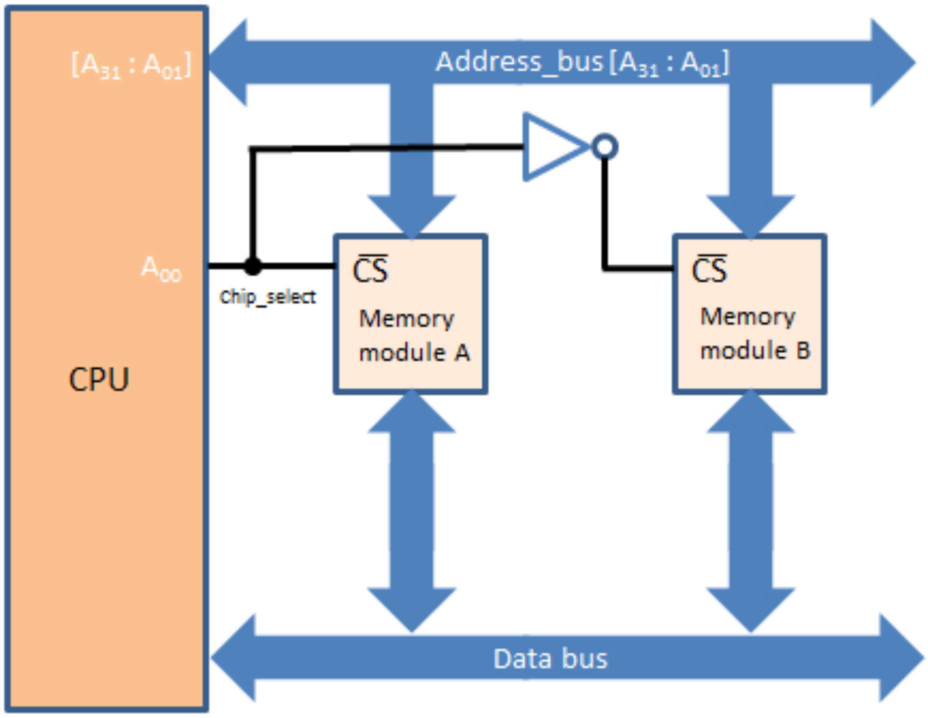
Address	Contents
0x80000	0x47
0x80001	0x87
0x80002	0x76
0x80003	0x34

Show, as an 8-digit hex value, the contents of register $t0 produced by each of the instructions listed below assuming first that the byte addressable system employs big endian storage order and then that it employs little endian storage order. Register $t1 contains the value 0x80000, **lw** is load word, **lh** is load half word, **lhu** is load half word unsigned, **lb** is load byte, and **lbu** is load byte unsigned.

	Big endian	Little endian
lw $t0,0($t1)	0x47877634	0x34768747
lh $t0,0($t1)	0x00004787	0xFFFF8747
lb $t0,3($t1)	0x00000034	0x00000034
lbu $t0,1($t1)	0x00000087	0x00000087
lh $t0,2($t1)	0x00007634	0x00003476
lb $t0,1($t1)	0xFFFFFF87	0xFFFFFF87

The instructions lb and lh sign extend the operand throughout the result register, while lbu and lhu perform zero-extension.

5. The memory system shown below consists of two modules (A and B) each of which contains 2147483648 storage cells. Each storage cell is 16 bits wide. For identification purposes, the cells within each module are numbered starting from 0. This system allows unaligned memory accesses.



The chip select (CS) is active low. That is CS=0 to select a chip.

a) Supply the missing operands for the two li instructions in the sequence that follows such that, when executed, the instruction sequence would store the pattern 0xCAFE into cell 5 within memory module A without affecting any other cells:

```
li      $t1, 0x0A ____
li      $t2, 0xCAFE __
sh      $t2,0($t1)
```

A_{00}, the LSB of the address, selects which module is accessed. The remaining address bits, A_{31} through A_{01} determine which cell within the selected module is accessed. A_{00} must be 0 to select module A and 1 to select module B since the chip select is active low. Hence the address required to access cell 5 within module A is 00000000000000000000000000001010 = 0x0A.

b) Supply the missing operands for the two li instructions in the sequence that follows such that, when executed, the instruction sequence would store the pattern 0xBABE into cell 7 within memory module B without affecting any other cells:

```
li      $t1, 0x0F _
li      $t2, 0xBABE __
sh      $t2,0($t1)
```

The address of cell 7 within module B is 00000000000000000000000000001111 = 0x0F.

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 = $2^{31}/2^{24} = 2^7 = 128$.

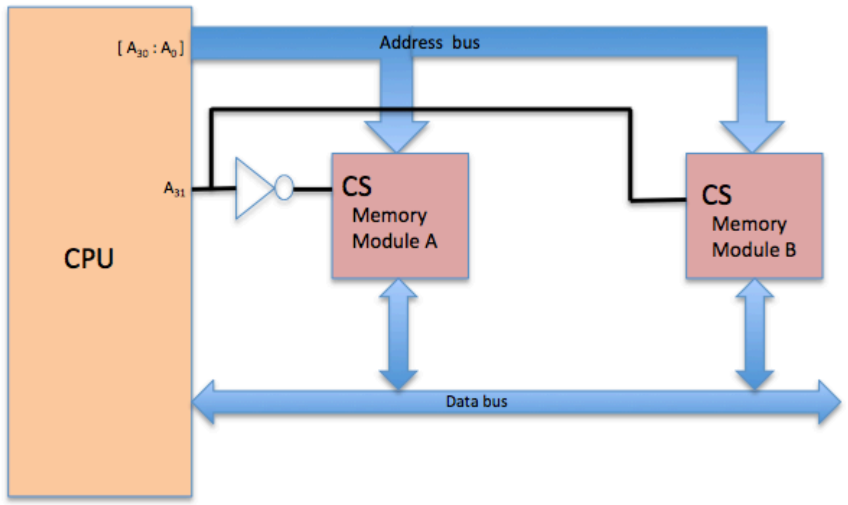
b) How would the bits within each 32-bit memory address be interpreted?

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.

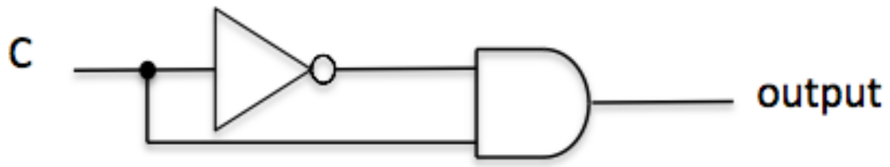
II. lw $t4,0($t2)
The contents of the 4-byte word starting at address 0x80000052 are read.
This address maps to module B and bits A30 through A1 =
00000000000000000000000010100, so cell 20 is read. The contents of cell 20 =
0xD3D2D1D0 and corresponds to bytes 80 through 83.
The low 2 bits = binary 10 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.

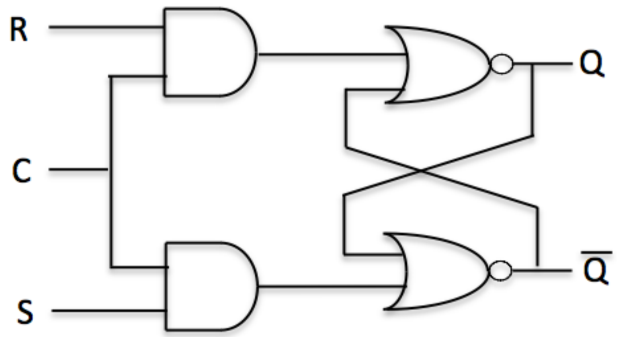
Example Set 5

1. The time interval required for an input signal to travel through a logic gate and appear on the output line is referred to as the gate "propagation delay". A pulse in a signal refers to a change from low to high and back to low. The time required for this low-to-high-to-low transition is the pulse width. Suppose that each gate in the circuit below has a propagation delay of 2ns. If a 4 ns wide positive pulse is applied to the C input, what would be the width of the corresponding pulse output from the circuit?



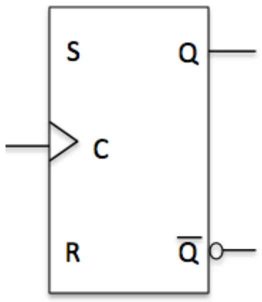
Answer:
C starts out as 0, so the output of the inverter starts at 1. The output of the AND gate starts at 0 AND 1 = 0.
C changes from 0 to 1, it takes 2ns before the output of the inverter to change to 0. During that time both AND gate inputs are 1, so the AND gate will output 1. Once inverter's output changes to 0, the AND gate outputs 0 and remains 0 even when C changes back to 0. Hence the output is a positive pulse and the circuit serves a positive pulse transition detector. Pulses are generated each time C goes from 0 to 1 (i.e., at each positive or leading clock edge).
(Note that such a pulse can be used as the dynamic clock input to cause a flip-flop to change state.)

2. Show an implementation of an SR latch that only changes state while enabled by a clock input.
Answer:



When the C (clock) input is 0, both AND gate output 0, so no state change occurs. When C=1, R as well as S are allowed to pass through the AND gates so the latch is set if S=1 and R=0. It is reset if R=1 and S=0. The output is unpredictable if S and R are both 1.

3. How does the behavior of an SR latch differ from that of an SR flip-flop?
The output from an SR flip-flop only changes at a clock edge while that of an SR latch can change whenever the clock enable is active (i.e., high). The block diagram for an edge triggered SR flip-flop is shown below.

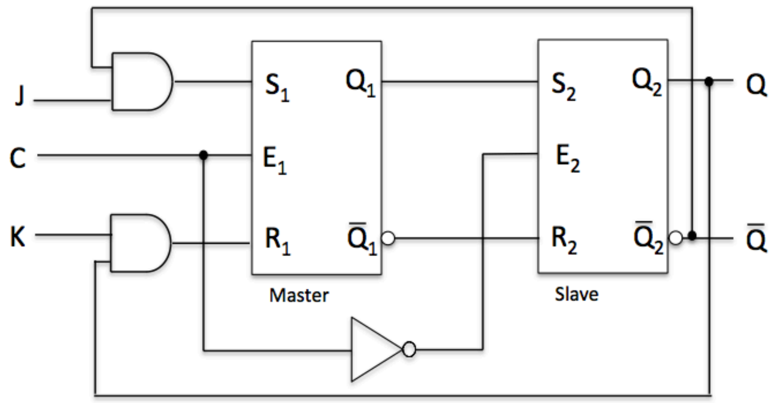


S-R Flip-flop

The trianglar symbol means that C is a dynamic input.

4. The output from an SR flip-flop is unpredictable when both S and R are 1. How can this unusable combination of inputs be made useful.
One way is to place an inverter between the S and R inputs so that they can never both be 1 at the same time. This produces a D flip-flop in which the inputs that were S and R and now derived from a single D input.

An alternative solution is to feed back the Q and Q' outputs to produce what is known as a J-K flip-flop shown below:



The Characteristic table for the J-K flip-flop is shown below:

INPUTS		OUTPUTS		Comments
J	K	Q	Q'	
0	0	Q_0	Q_0'	No change
0	1	0	1	RESET
1	0	1	0	SET
1	1	Q_0'	Q_0	toggle

Each latch reacts to its inputs only when enabled.
Q_0 represents the previous output.

When C is high the master is enabled and responds to the signals from the AND gates. The inverter disables the slave when C is high.
Assume that initially the Q output is 0, so Q'=1.
If J and K are both 0, each AND gate outputs 0 so there is no change in the outputs.
If J =1 and K=0, Q1 is set to 1 (since S1 = 1 AND 1 = 1) and Q1'=0
When C goes back to 0, the slave is enabled and has S2=Q1=1, R2=Q1'=0 so Q2 is set to 1

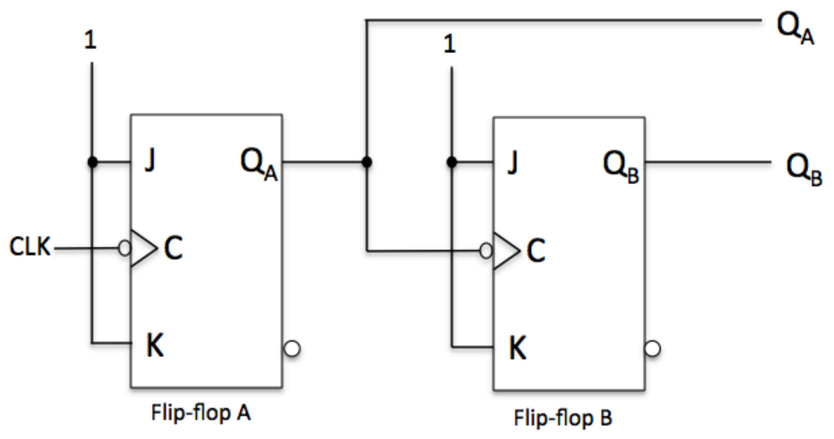
Assume the 1 state, that is, Q=1.
If J=0 and K=1, Q1 is reset to 0 (since R1 = 1 AND 1 = 1) and Q1'=1.
When C goes back to 0, the slave is enabled and has S2=Q1=0, R2=Q1'=1 so Q2 is reset to 0.

Assume the 0 state (Q=0).
If both J=1 and K=1, S1 = J AND Q' = 1 AND 1 = 1; R1= K AND Q = 1 AND 0 = 0
When the clock goes back to 0, the slave is enabled and has S2=1, R2=0 so the output toggles from 0 to 1.

Assume the 1 state(Q=1).
If both J=1 and K=1, S1 = J AND Q' = 1 AND 0 = 1; R1 = K AND Q = 1 AND 1 = 1
When the clock goes back to 0, the slave is enabled and has S2=0, R2=1 so the output toggles from 1 to 0.

5. What is an example of an application of J-K flip-flops with both J and K set to 1?
One application is in the implementation of a binary counter in which each bit in the value is generated by a separate flip-flop. As each bit position increments, it changes from 0 to 1 or from 1 to 0 (i.e., it toggles).
Another is in producing a frequency divider. Since the output only changes on either the leading or trailing clock edge, the rate at which the Q output changes will be one half the rate at which the flip-flop is clocked (i.e., the frequency is divded by 2).

6. Shown below is a block diagram representing two J-K flip-flops.
The J and K inputs for both are tied to a value of 1. The Q output of flip-flop A is used to clock or activate flip-flop B. Assume that the output from each flip-flop is initially 0. Show the outputs Q_A and Q_B for the first 4 clock transitions.



Recall that where lines cross there is no connection unless a dot is present.

The bubble on the clock input denotes that the flip-flop changes on the falling or trailing clock edge when the clock goes from 1 to 0.

Initially Q_B and Q_A = 0 0
At the first clock transition Q_A toggles from 0 to 1 but the second flip-flop does not react since its clock input is Q_A (which went from 0 to 1, not 1 to 0).
Hence Q_B and Q_A = 0 1.

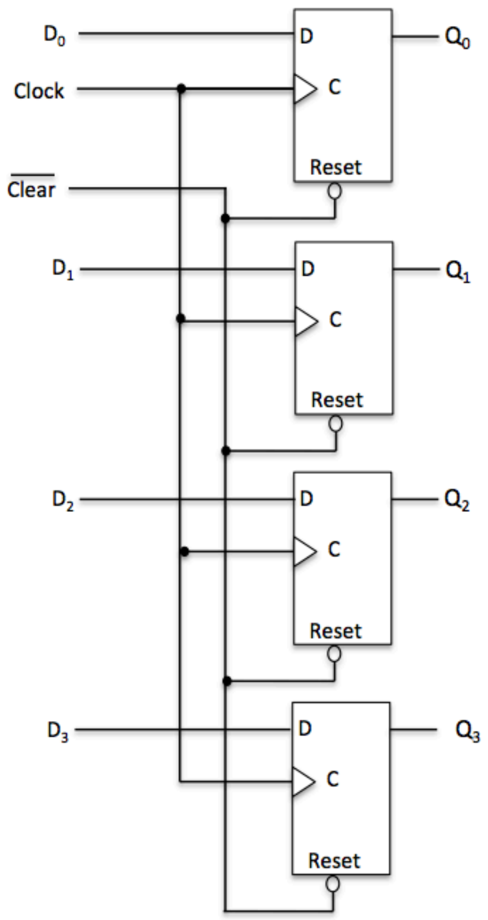
At the second clock transition Q_A toggles from 1 to 0 and causes flip-flop B to toggle Q_B from 0 to 1. Hence Q_B and Q_A = 1 0.

At the third clock transition Q_A toggles from 0 to 1 but the second flip-flop does not react. Hence Q_B and Q_A = 1 1.

At the 4th clock transition Q_A toggles from 1 to 0 and causes flip-flop B to toggle Q_B from 1 to 0. Hence Q_B and Q_A = 0 0.
Therefore the behavior is that of a 2-bit binary counter that increments by 1 for each clock transition and rolls back to 00 from 11. More flip-flops could be included to create a binary counter with the desired number of bits.

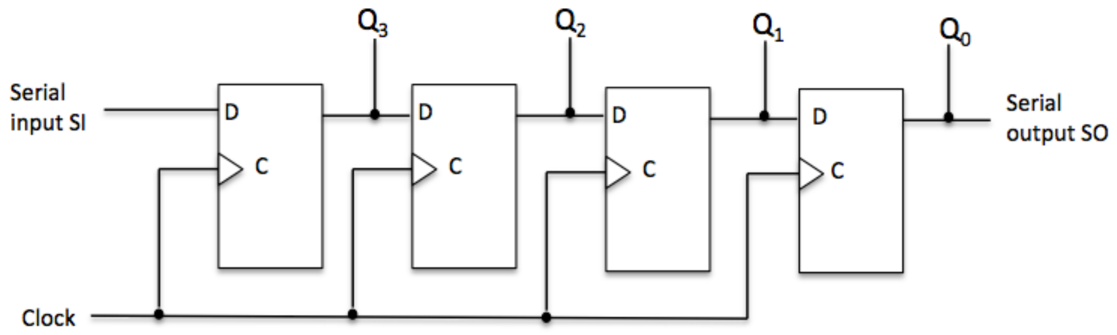
7. Show how a 4-bit register could be implemented using D flip-flops.

Answer:
One implementation is shown below:



The Clear signal when set to 0 clears the 4-bit register (i.e., to reset the register). When the clock makes a positive transition (from 0 to 1) each of the 4 data inputs is stored into a separate D flip-flop. The Q outputs show the current contents of the register.

8. Show how D flip-flops can be used to create a 4-bit serial shift register.
Answer:
One simple implementation is shown below:



This is a chain of flip-flops in cascade, with the output of one flip-flop connected to the input of the next flip-flop on the right. All flip-flops are controlled by a common clock pulse. For each clock pulse the current serial input SI is loaded into the leftmost flip-flop; the output of each flip-flop is loaded into its neighbor on the right and the output of the rightmost flip-flop appears on SO (the serial out). Hence the behavior is that of a right shift. After four clock pulses, the register is filled and the 4 bits contained in the register can be read as $Q_3Q_2Q_1Q_0$. The original 4 bits in the register will have been serially shifted out (i.e., one at a time) through SO.

9 How can a J-K flip-flop be transformed into a D flip-flop.
This can be done by inserting an inverter between the J and K inputs. With this change the output will be set to 1 when J=1 and the output will be reset to 0 when J=0 (since K=J' = 1). So the flip-flop behaves like D flip-flop with J as the D input.

10. A T flip-flop (toggle flip-flop) is one that has a single input T and complements its output each time that T makes a transition from 0 to 1. How can a J-K flip-flop be transformed into a T flip-flop?
If the J and K inputs are connected together the desired behavior will be obtained. This makes J and K equal. When the input is 1, J=1 and K=1 which causes the output to toggle. If the input is 0, J=0 and K=0 so there is no change in the output.