Solved Problems

1. Suppose one has the following memory map as a result of a core dump. The memory is byte addressable.

Address	0x200	0x201	0x202	0x203
Contents	02	04	06	08

What is the value of the 32–bit long integer stored at address 0x200?

This is stored in the four bytes at addresses 0x200, 0x201, 0x202, and 0x203.

Big Endian: The number is 0x02040608, or 0204 0608. Its decimal value is
$2 \bullet 256^3 + 4 \bullet 256^2 + 6 \bullet 256^1 + 8 \bullet 1 = 33,818,120$

Little Endian: The number is 0x08060402, or 0806 0402. Its decimal value is
$8 \bullet 256^3 + 6 \bullet 256^2 + 4 \bullet 256^1 + 2 \bullet 1 = 134,611,970.$

NOTE: Read the bytes backwards, not the hexadecimal digits.
Powers of 256 are $256^0 = 1$, $256^1 = 256$,
$256^2 = 65536$, $256^3 = 16,777,216$

2. Suppose one has the following memory map as a result of a core dump. The memory is byte addressable.

Address	0x200	0x201	0x202	0x203
Contents	02	04	06	08

What is the value of the 16–bit integer stored at address 0x200?

This is stored in the two bytes at addresses 0x200 and 0x201.

Big Endian The value is 0x0204.
The decimal value is $2 \bullet 256 + 4 = 516$

Little Endian: The value is 0x0402.
The decimal value s $4 \bullet 256 + 2 = 1,026$

Note: The bytes at addresses 0x202 and 0x203 are not part of this 16–bit integer.

3. You are asked to implement a 128M by 32 memory (1M = 2^{20}), using only 16M by 8 memory chips.
   a) What is the minimum size of the MAR?
   b) What is the size of the MBR?
   c) How many 16M by 8 chips are required for this design?

Answer: a) $128M = 2^7 \bullet 2^{20} = 2^{27}$, so the minimum MAR size is **27 bits**.
b) The MBR size is **32 bits**.
c) $128M \bullet 32 / 16M \bullet 8 = 8 \bullet 4 =$ **32 chips**.

4. Complete the following table describing the memory and chip count needed to fabricate.

Memory System Capacity	Number of bits in MAR	Number of bits in MBR	Number of Chips Needed if the capacity of each chip is		
			1K by 4	2K by 1	1K by 8
64K by 4					
64K by 8					
32K by 4					
32K by 16					
32K by 32					
10K by 8					
10K by 10					

ANSWER: We begin by showing the general formulae and then giving a few specific answers. First, we must define some variables, so that we may state some equations.

Let N1 be the number of addressable units in the memory system
M1 be the number of bits for each entry in the memory system
N2 be the number of addressable units in the memory chip
M2 be the number of bits for each entry in the memory chip.

So that for making a 64K by 4 memory from a 1K by 8 chip, we have

$N1 = 64K = 2^6 \bullet 2^{10} = 2^{16}$, as $1K = 2^{10} = 1{,}024$.
$M1 = 4$
$N2 = 1K = 2^{10}$.
$M2 = 8$.

Number of bits in MAR and MBR

These numbers are defined by the memory system parameters and have nothing to do with the memory chips used to construct the memory. For a N1 by M1 memory system, we have

P bits in the MAR, where $2^{P-1} < N1 \le 2^P$.
M1 bits in the MBR.

Note that in most modern computers, the actual number of bits in the MAR is set at design time and does not reflect the actual memory size. Thus, all computers in the Pentium™ class have 32-bit MAR's, even if the memory is $256MB = 256 \bullet 1MB = 2^8 \bullet 2^{20}B = 2^{28}$ bytes.

$N1 = 32K = 2^5 \bullet 2^{10} = 2^{15}$. Solve $2^{P-1} < 2^{15} \le 2^P$ to get P = 15, or 15 bits in the MAR.
$N1 = 64K = 2^6 \bullet 2^{10} = 2^{16}$. Solve $2^{P-1} < 2^{16} \le 2^P$ to get P = 16, or 16 bits in the MAR.
$N1 = 10K = 5 \bullet 2K = 5 \bullet 2^{11} = 1.25 \bullet 2^{13}$, not a power of 2.
Solve $2^{P-1} < 1.25 \bullet 2^{13} \le 2^P$ to get P = 14. Note that $2^{13} = 8K$ and $2^{14} = 16K$.

With this much, we may start filling in the table.

Memory System Capacity	Number of bits in MAR	Number of bits in MBR	Number of Chips Needed if the capacity of each chip is		
			1K by 4	2K by 1	1K by 8
64K by 4	16	4			
64K by 8	16	8			
32K by 4	15	4			
32K by 16	15	16			
32K by 32	15	32			
10K by 8	14	8			
10K by 10	14	10			

For most of the table, one may compute the number of chips needed by the following formula: Chips = (N1 • M1) / (N2 • M2), or the total number of bits in the memory system divided by the total number of bits in the memory chip. In actual fact, this works only when one of the two following conditions holds:

either M1 / M2 is a whole number (as M1 = 4 and M2 = 1),
or M2 / M1 is a whole number (as M1 = 4 and M2 = 8).

The analysis in the 10K-by-10 case, in which neither of these conditions holds, is a bit more complicated. Here we present a detailed discussion of the 64K-by-4 case, followed by the answers to all but the 10K-by-10 case, which we also discuss in detail.

For 64K-by-4 fabricated from 1K-by-4, it is obvious that each 4-bit entry in the memory system is stored in one 4-bit memory chip, so that the total number of chips required is simply 64, or (64K • 4) / (1K • 4).

For 64K-by-4 fabricated from 2K-by-1 chips, it should be obvious that four entries in the 2K-by-1 chip are required to store each of the 4-bit entries in the memory system. The easiest way to achieve this goal is to arrange the memory chips in "banks", with four of the chips to each bank. The number of banks required is 64K / 2K = 32, for a total of 128 chips. Note that this agrees with the result (64K • 4) / (2K • 1) = 256K / 2K = 128.

For 64K-by-4 fabricated from 1K-by-8 chips, it should be obvious that the 8-bit entries in the chip can store two of the 4-bit entries in the memory system. For this reason, each 1K-by-8 chip can store 2K entries in the main memory and the number of chips needed is 64K / 2K or 32. This answer is the same as (64K • 4) / (1K • 8) = 256K / 8K = 32.

From this point until the 10K-by-8 entry we may just argue relative sizes of the memories, so that the 64K-by-8 memory is twice the size of the 64K-by-4, the 32K-by-4 memory is half the size of the 64K-by-4 memory, etc.

We now present the table to this point.

Memory System Capacity	Number of bits in MAR	Number of bits in MBR	Number of Chips Needed if the capacity of each chip is		
			1K by 4	2K by 1	1K by 8
64K by 4	16	4	64	128	32
64K by 8	16	8	128	256	64
32K by 4	15	4	32	64	16
32K by 16	15	16	128	256	64
32K by 32	15	32	256	512	128
10K by 8	14	8			
10K by 10	14	10			

10K-by-8 memory

From 1K-by-4

$(N1 \bullet M1) / (N2 \bullet M2) = (10K \bullet 8) / (1K \bullet 4) = 80K / 4K = 20$

From 2K-by-1

$(N1 \bullet M1) / (N2 \bullet M2) = (10K \bullet 8) / (2K \bullet 1) = 80K / 2K = 40$

From 1K-by-8

$(N1 \bullet M1) / (N2 \bullet M2) = (10K \bullet 8) / (1K \bullet 8) = 80K / 8K = 10$

10K-by-10 memory

From 2K-by-1

$(N1 \bullet M1) / (N2 \bullet M2) = (10K \bullet 10) / (2K \bullet 1) = 100K / 2K = 50$

We run into trouble with the 1K-by-4 chips because 10/4 = 2.5; thus 4 does not divide 10. The problem is how to spread two entries (20 bits) over five chips and retrieve the 10-bit words efficiently. It cannot be done; one must allocate three chip entries per 10-bit word.

From 1K-by-4

We solve $(N1 / N2) \bullet \lceil M1 / M2 \rceil = (10K / 1K) \bullet \lceil 10 / 4 \rceil = 10 \bullet 3 = 30$.

From 1K-by-4

We solve $(N1 / N2) \bullet \lceil M1 / M2 \rceil = (10K / 1K) \bullet \lceil 10 / 8 \rceil = 10 \bullet 2 = 20$.

Memory System Capacity	Number of bits in MAR	Number of bits in MBR	Number of Chips Needed if the capacity of each chip is		
			1K by 4	2K by 1	1K by 8
64K by 4	16	4	64	128	32
64K by 8	16	8	128	256	64
32K by 4	15	4	32	64	16
32K by 16	15	16	128	256	64
32K by 32	15	32	256	512	128
10K by 8	14	8	20	40	10
10K by 10	14	10	30	50	20

5. A 256-word memory has its words numbered from 0 to 255. Define the address bits for each of the following. Each address bit should be specified as 0, 1, or d (don't-care).
   a) Word 48
   b) Lower half of memory (words 0 through 127)
   c) Upper hald of memory (words 128 through 255)
   d) Even memory words (0, 2, 4, etc.)
   e) Any of the eight words 48 through 55.

ANSWER: Note that $2^8 = 256$, so we need an 8-bit address for this memory.

Recall that memory addresses are unsigned integers, so that we are dealing with unsigned eight-bit integers here. The bit values are given in this table.

Bit	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
Value	128	64	32	16	8	4	2	1

So, in eight-bit unsigned binary we have decimal 128 = 1000 0000 in binary, and decimal 127 = 128 – 1 = 0111 1111 in binary.

a) Word 48.
   The answer here is just the binary representation of decimal 48.
   Since 48 = 32 + 16, the answer is **0011 0000**.

   NOTE: This is an eight-bit number, so it needs two leading 0's.

b) Words 0 through 127.
   We represent each address and then determine what bits are common.
   Decimal 0 = 0000 0000, and
   Decimal 127 = 0111 1111, so the answer is **0ddd dddd**.
   Any address beginning with a 0 is in the lower half of memory.

c) Words 128 through 255.
   Decimal 128 = 1000 0000 (unsigned 8-bit binary), and
   Decimal 255 = 1111 1111, so the answer is **1ddd dddd**.

d) Even memory words.
   Each even number has its least significant bit equal 0, as the LSB is the "1 bit". So the answer is **dddd ddd0**.

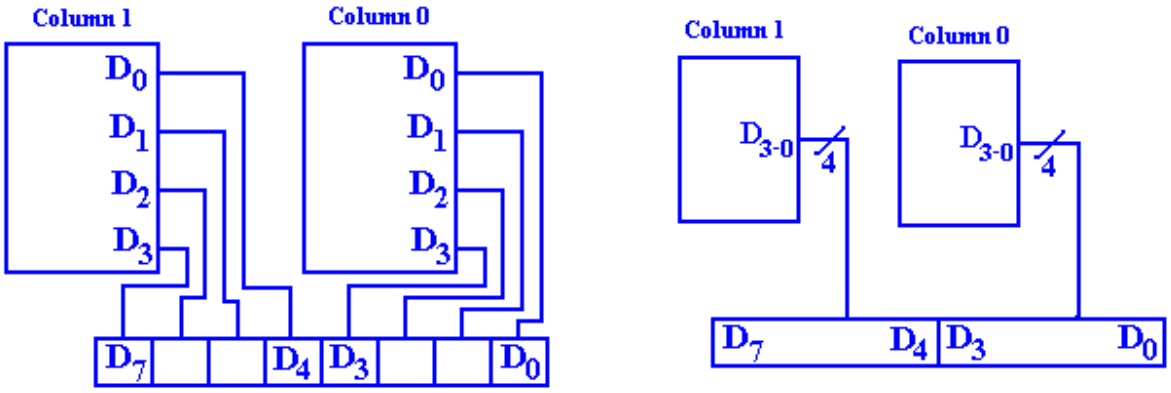
e) Any of the eight words 48 through 55.
   Decimal 48 = 0011 0000 (see answer a), and
   Decimal 55 = 0011 0111 as 55 = 48 + 7,
   We look for the common bits and determine the answer as **0011 0ddd**.

6 Design a 16K-by-8 memory using 2K-by-4 memory chips. Arrange the chips in an 8 by 2 array (8 rows of 2 columns) and design the decoding circuitry.

ANSWER: The first part of the solution is to define what we are given and what we are asked to build. From that, we can develop the solution, introducing a new term.

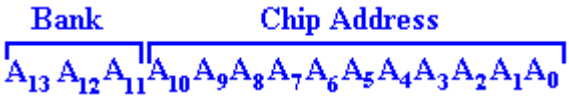
We are given a collection of 2K-by-4 memory chips. As $2K = 2^{11}$, each of these chips has eleven address lines, denoted $A_{10}A_9A_8A_7A_6A_5A_4A_3A_2A_1A_0$, four data lines, $D_3D_2D_1D_0$, and the required control and select lines. We are asked to design a 16K-by-8 memory.
As $16K = 2^{14}$, this memory has 14 address lines, denoted A_{13} through A_0. The memory to be designed has eight data lines, denoted D_7 through D_0, and necessary control lines.

Because we are using 4-bit chips to create an 8-bit memory, we shall arrange the chips in rows of two, with one chip holding the high-order four bits and the other the low-order four bits of the eight-bit entry. Each such row is called a **bank** of memory. The end result of this design is the arrangement of memory into a number of rows and two columns. The figure below shows the use of the columns. The drawing on the left shows four data lines from each chip feeding four bits in the eight-bit MBR. The drawing at the right shows a common short-hand for expressing the same drawing.

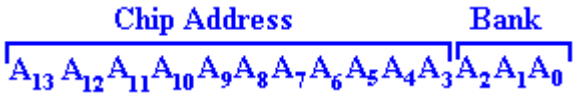


The design suggested above divides the memory into banks of 2K-by-8, so that we need 16K / 2K = 8 banks for the memory to be designed. This suggests that we need 8 banks of 2 chips each for a total of 16 chips. Using the analysis of problem 3.1, we establish the number of chips needed as (16K • 8) / (2K • 4) = 128K / 8K = 16, as expected.

If we have 8 banks, each bank containing 2K addresses, we need 3 bits ($8 = 2^3$) to select the bank and 11 bits ($2K = 2^{11}$) to address each chip in the bank. We have stipulated that the 16K memory requires 14 address lines, as $16K = 2^{14}$, so we have exactly the right bit count. There are two standard ways to split the 14-bit address into a bank select and chip address.



Banked Memory



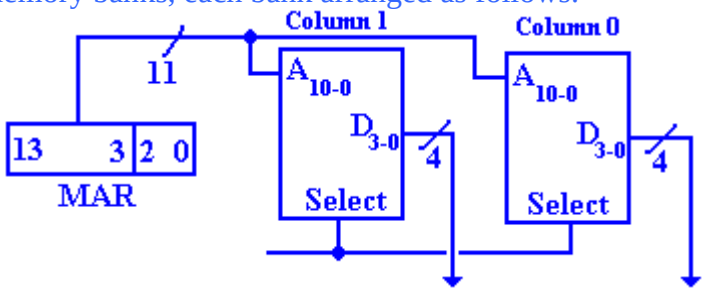
Interleaved Memory

The observant student will note that the number of ways to split the 14-bit address into a 3-bit bank number and an 11-bit chip address is exactly

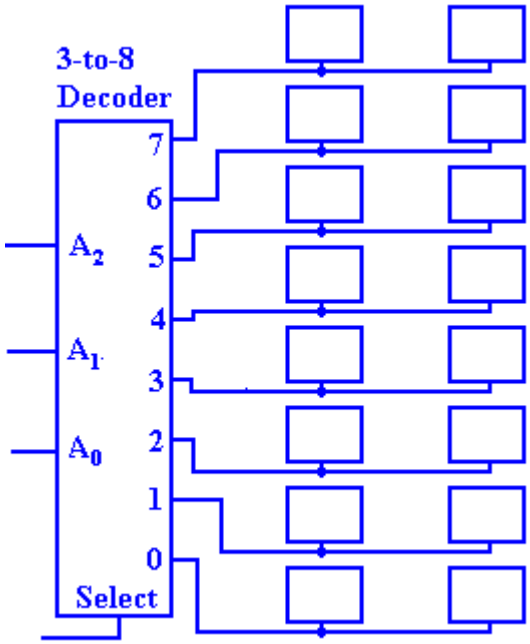
$$\binom{14}{3} = \frac{14 \cdot 13 \cdot 12}{3 \cdot 2 \cdot 1} = 364$$

but that only the two methods mentioned above make any sense.

The solution presented in these notes will be based on the **interleaved** memory model. The basis of the design will be a set of memory banks, each bank arranged as follows.



Note that the eleven high-order bits of the system MAR are sent to each chip in every memory bank. Note that in each memory bank, both chips are selected at the same time. We now use a 3-to-8 decoder to select the memory bank to be accessed.

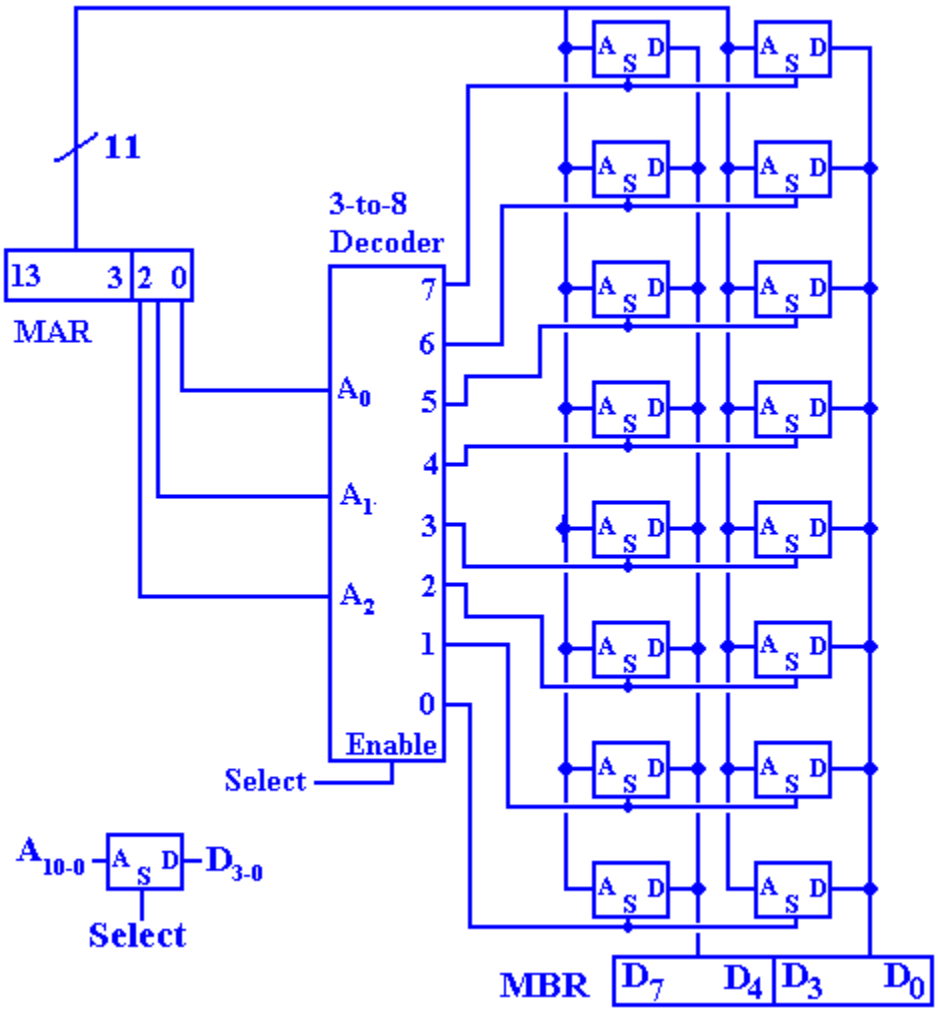


The circuit at left shows the basic layout for the chip select part of the addressing. This diagram assumes interleaved memory, in which address bits $A_2A_1A_0$ serve to select the memory bank that holds the addressed byte and address bits $A_{13}A_{12}A_{11}A_{10}A_9A_8A_7A_6A_5A_4A_3$ are passed as an 11-bit address to each of the sixteen chips.

Again, it would be equally valid to have address bits $A_{13}A_{12}A_{11}$ as input to the decoder and the low-order 11 bits passed to each chip. This second arrangement is not interleaved memory.

We now have all of the components of a complete design, so now let's show the whole thing.

Here is the complete design of the 16K-by-8 memory as fabricated from 2K-by-4 chips.



The chip in the lower left explains the notation used in the main drawing.
For each of the sixteen memory chips, we have the following

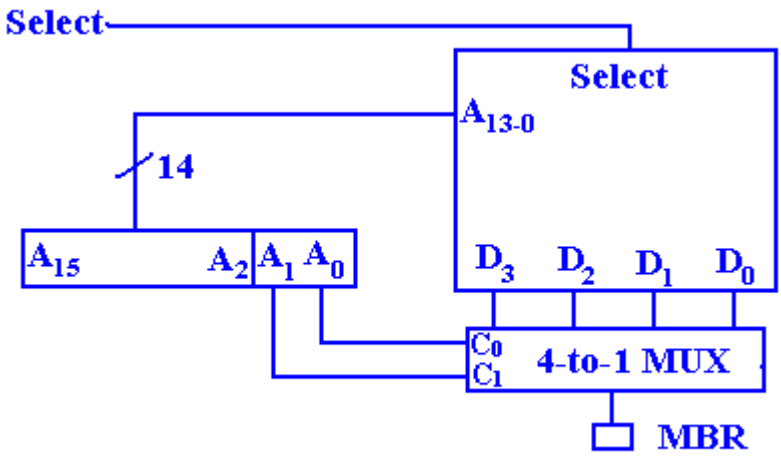
A	the eleven address inputs $A_{11\text{-}0}$.
D	the four data I/O lines $D_3D_2D_1D_0$.
S	the chip select – active high.

7 You are given a 16K-by-4 ROM unit. Convert it to a 64K-by-1 ROM. Treat the 16K-by-4 ROM as one unit that you cannot alter. Only logic external to this 16K-by-4 ROM unit is to be used or shown.

ANSWER: This problem is almost opposite the previous problem. We are given a ROM (Read-Only Memory) that is 16K-by-4. As $16K = 2^{14}$, the address register for this chip must have 14 bits: $A_{13\text{-}0}$. The data buffer of this memory has four bits: $D_3D_2D_1D_0$.

The problem calls for the design of a 64K-by-1 ROM. As $64K = 2^{16}$, this memory has a 16-bit address for each bit. Of the 16-bit address that goes to this memory, 14 bits must go to the 16K-by-4 memory chip and 2 bits used to select one of the four bits output from the chip.

The only other task is to use a 4-to-1 multiplexer to select which of the 4 bits from the 16K-by-4 chip is to be used.



8 A given computer design calls for an 18-bit MAR and an 8-bit MBR.
a) How many addressable memory elements can the memory contain?
b) How many bits does each addressable unit contain?
c) What is the size of the address space in bytes?

NOTE: The answer can be given in many ways. If the answer were 16K, you might say either 16K, 2^{14}, or 16384.

ANSWER: Recall that $2^{18} = 2^8 \bullet 2^{10} = 256K = 256 \bullet 1024 = 262,144$.

a) As the MAR contains 18 bits, the maximum number of addressable units is 2^{18} or 256K. The answer 262,144 is correct, but not favored.
b) As the MBR contains 8 bits, so does each addressable unit. The addressable unit is an 8-bit byte.
c) As a result of the above two answers, the address space is 256KB or 2^{18} bytes.

9 You are asked to design a 128K-by-16 memory using available chips.
How many 16K-by-4 memory chips would be required for the design?

ANSWER: $(128K \bullet 16) / (16K \bullet 4) = 2048K / 64K = 32$
Equivalently $(128K / 16K) \bullet (16 / 4 = 8 \bullet 4 = 32$

10 You are given a number of chips and asked to design a 64KB memory.
You immediately realize that this requires 16 address lines, as $64K = 2^{16}$. The memory is interleaved, with 12 address bits ($A_{15} – A_4$) going to each byte-addressable memory chip and 4 address bits ($A_3A_2A_1A_0$) going to select the memory chip.

a) What is the size of each chip in bytes (or KB)?
b) How many chips are found in this design?

ANSWER: The memory chip is byte-addressable, implying that each byte in the chip has a distinct address and that the number of addresses in the chip equals the number of bytes.

a) There are 12 address bits for each of the memory chips,
so each memory chip has 2^{12} bytes = $2^2 \bullet 2^{10}$ bytes = $4 \bullet 2^{10}$ bytes = 4KB.
b) There are 4 address lines used to select the memory chips,
so there must be $2^4 = 16$ chips.

NOTE: The key idea for this and the previous question is that P bits will select 2^P different items, alternately that a P-bit unsigned binary number can represent decimal numbers in the range 0 to $2^P – 1$ inclusive, often written as the closed interval $[0, 2^P – 1]$. As an example, an 8-bit binary number will represent unsigned decimal numbers in the range 0 to $2^8 – 1$, or 0 to 255, while a 16-bit binary number will represent unsigned decimal numbers in the range $[0, 2^{16} – 1]$ or [0, 65535].

Another way of saying the same thing is that the representation of a positive integer N requires P bits where $2^{P–1} < N \le 2^P$. This simple idea is important in many areas of computer science, but seems to elude some students.

11 You are asked to implement a 128M by 32 memory ($1M = 2^{20}$),
using only 16M by 8 memory chips.

a) What is the minimum size of the MAR?
b) What is the size of the MBR?
c) How many 16M by 8 chips are required for this design?

ANSWER: Remember that $128M = 2^7 \bullet 2^{20} = 2^{27}$ and that $16M = 2^{24}$.

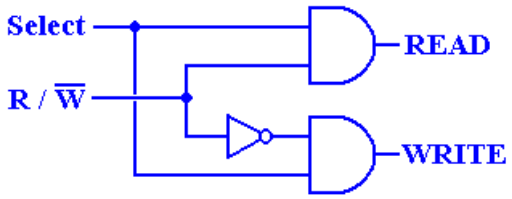
a) To address $128M = 2^{27}$ entries, we need a **27-bit MAR**.
b) Since each entry is 32-bits, we have a **32-bit MBR**.
c) The number of chips is $(128M \bullet 32) / (16M \bullet 8)$, which evaluates to
$(2^{27} \bullet 2^5) / (2^{24} \bullet 2^3) = 2^{32} / 2^{27} = 2^5 =$ **32**.

12 A CPU outputs two control signals: Select and $R / \overline{W}$, interpreted as follows.
If Select = 0, the memory does nothing.
If Select = 1, the memory is read if $R / \overline{W}$ = 1, otherwise it is written.

Draw a circuit to convert these two signals into the following control signals
READ If READ = 1, the CPU reads from memory.
WRITE If WRITE = 1, the CPU writes to memory.
Insure that READ = 1 and WRITE = 1 cannot occur at the same time.

ANSWER: Note, if Select = 0, then both READ = 0 and WRITE = 0.
if Select = 1 and $R / \overline{W}$ = 1, then READ = 1 and WRITE = 0
if Select = 1 and $R / \overline{W}$ = 0, then READ = 0 and WRITE = 1.

The circuit follows from those observations.



13 A computer has a 24-bit MAR. Each 16-bit word is individually addressable.
All answers larger than 128 should be given as a power of two.
a) How many words can this computer address?
b) Assuming 8-bit bytes, how many bytes can this computer address?
c) What is the size of the MBR? This is not a trick question.

ANSWERS: a) The MAR size is 24 bits; the computer can address **2^{24} words.**
This is also $2^4 \bullet 2^{20}$ words = $16 \bullet 2^{20}$ words = 16 M words.

b) One 16-bit word = 2 bytes, so the memory size is $2 \bullet 2^{24}$ bytes or **2^{25} bytes.**
This is also $2^5 \bullet 2^{20}$ bytes = $32 \bullet 2^{20}$ bytes = 32 M bytes.

c) The MBR is 16 bits in size, the same as the size of the addressable unit.

14 Examine the following memory map of a byte–addressable memory.
Let W refer to address 109.

107	108	109	10A	10B
00	11	23	17	CA

a) Assuming big–endian organization, what is the value of the 16–bit integer at W?
b) Assuming little–endian organization, what is the value of the 16–bit integer at W?

Answer: The 16–bit entry at address 109 occupies bytes 109 and 10A, as follows.

107	108	**109**	**10A**	10B
00	11	**23**	**17**	CA

a) In big–endian, the "big end" is stored first: 2317

b) In little–endian, the "little end" is stored first 1723.

15 A computer has a cache memory set–up, with the cache memory having an access time of 6 nanoseconds and the main memory having an access time of 80 nanoseconds. This question focuses on the effective access time of the cache.
a) What is the minimum effective access time for this cache memory?
b) If the effective access time of this memory is 13.4 nanoseconds, what is the hit ratio?

ANSWER:

a) The cache memory cannot have an access time less than that of the cache, so **6 nsec**.

b) The equation of interest here is $T_E = h \bullet T_P + (1 - h) \bullet T_S$. Using the numbers, we have $13.4 = h \bullet 6 + (1 - h) \bullet 80$, or $13.4 = 80 - 74 \bullet h$, or $74 \bullet h = 66.6$, or **h = 0.9**.

16. (20 points) Suppose a computer using direct mapped cache has 2^{32} words of main memory and a cache of 1024 blocks, where each cache block contains 32 words.

a) How many blocks of main memory are there?

b) What is the format of a memory address as seen by the cache, that is, what are the sizes of the tag, block, and word fields?

c) To which cache block will the memory reference 0000 63FA map?

ANSWER: Recall that $1024 = 2^{10}$ and $32 = 2^5$, from which we may conclude $1024 = 32 \bullet 32$.

a) The number of blocks is $2^{32} / 2^5 = 2^{(32-5)} = \mathbf{2^{27}} = 2^7 \bullet 2^{20} = 128$ M = 134, 217, 728.

b) 32 words per cache line $\Rightarrow$ Offset address is **5 bits**.
1024 blocks in the cache $\Rightarrow$ Block number is **10 bits**.
2^{32} words $\Rightarrow$ 32 bit address $\Rightarrow$ 32 – (5 + 10) = 10 – 15 = **17 bit tag**.

Bits	31		15	14		5	4		0
Contents	Tag			Block Number			Offset within block		

c) The number of bits allocated to the block number and offset are 15. We examine the last four hex digits of 0000 63FA, as they correspond to 16 binary bits.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Hex	6				3				F				A			
Binary	0	1	1	**0**	**0**	**0**	**1**	1	1	1	1	1	1	0	1	0
		11 0001 1111 = 0x31F											1 1010 = 0x1A = 26			

0x31F = 3•256 + 1•16 + 15 = 768 + 16 + 15 = **799**.

17 A given computer design calls for an 18-bit MAR and an 8-bit MBR.

a) How many addressable memory elements can the memory contain?

b) How many bits does each addressable unit contain?

c) What is the size of the address space in bytes?

ANSWER: Recall that $2^{18} = 2^8 \bullet 2^{10} = 256\text{K} = 256 \bullet 1024 = 262{,}144$.

a) As the MAR contains 18 bits, the maximum number of addressable units is 2^{18} or 256K. The answer 262,144 is correct, but not favored.

b) As the MBR contains 8 bits, so does each addressable unit. The addressable unit is an 8-bit byte.

c) As a result of the above two answers, the address space is 256KB or 2^{18} bytes.

18 A computer memory system uses a primary memory with 100 nanosecond access time, fronted by a cache memory with 8 nanosecond access time. What is the effective access time if
a) The hit ratio is 0.7?
b) The hit ratio is 0.9?
c) The hit ratio is 0.95?
d) The hit ratio is 0.99?

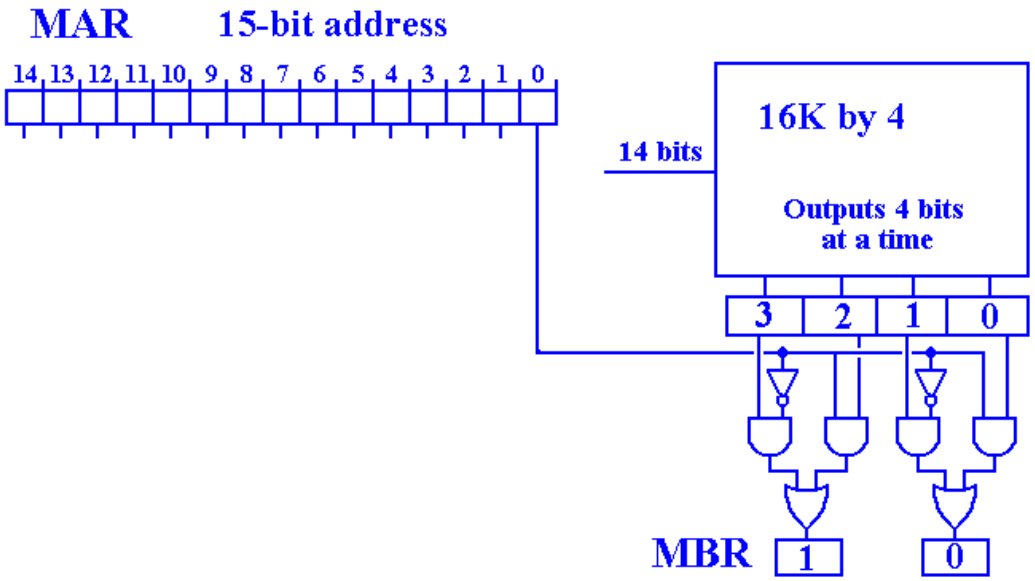
ANSWER: There is a problem with names here. For the cache scenario, we have
T_P as the access time of the cache
T_S as the access time of the primary memory.

The equation is $T_E = h \bullet 8 + (1 - h) \bullet 100$.

a) h = 0.7 $T_E = 0.7 \bullet 8 + (1 - 0.7) \bullet 100 = 0.7 \bullet 8 + 0.3 \bullet 100 = 5.6 + 30.0$ = **35.6**
b) h = 0.9 $T_E = 0.9 \bullet 8 + (1 - 0.9) \bullet 100 = 0.9 \bullet 8 + 0.1 \bullet 100 = 7.2 + 10.0$ = **17.2**
c) h = 0.95 $T_E = 0.95 \bullet 8 + (1 - 0.95) \bullet 100 = 0.95 \bullet 8 + 0.05 \bullet 100 = 7.6 + .5.0$ = **12.6**
d) h = 0.99 $TE = 0.99 \bullet 8 + (1 - 0.99) \bullet 100 = 0.99 \bullet 8 + 0.01 \bullet 100 = 7.92 + 1.0$ = **8.92**

19 (20 points) You are given a 16K x 4 ROM unit. Convert it to a 32K x 2 ROM. Use only AND, OR, and NOT gates and possible D flip–flops in your design. Treat the 16K x 4 ROM as one unit you cannot alter. Include only logic gates external to the ROM unit.

ANSWER:
The 16K by 4 ROM has $16K = 2^{14}$ addressable entries, each of 4 bits. We are to convert this to a 32K by 2 ROM, with $32K = 2^{15}$ addressable entries, each of 2 bits. The 15–bit address for the latter must be broken into a 14–bit address for the former and a selector.



20 The 16–bit number, represented in hexadecimal as 0xCAFE,
is to be stored at the address 0x40D in a byte–addressable memory.

a) What are the addresses of the two bytes associated with this 16–bit number?
b) What are the contents of these addresses if the number is stored in big–endian form?
c) What are the contents of these addresses if the number is stored in little–endian form?

ANSWER: a) The number is stored in addresses 0x40D and 0x40E.

b) In big–endian, the more significant byte is stored first.
0x40D 0xCA
0x40E 0xFE

c) In little–endian, the less significant byte is stored first.
0x40D 0xFE
0x40E 0xCA

21 Suppose a computer using fully associative cache has 2^{20} words of main memory and a cache of 512 blocks, where each cache block contains 32 bytes.
The memory is byte addressable; each byte has a distinct address.

a) How many blocks of main memory are there?
b) What is the format of a memory address as seen by the cache, that is, what are the sizes of the tag and word fields?

Answer: Recall that $32 = 2^5$.

a) The number of blocks is $2^{20} / 2^5 = 2^{(20-5)} = 2^{15} = 2^5 \bullet 2^{10} = 32\text{ K} = 32{,}768$

b) For associative mapping the address has only tag and word offset in the block.
32 byte block $\Rightarrow$ 5 bit offset.
20 bit address $\Rightarrow (20 - 5) = 15$ bit tag.

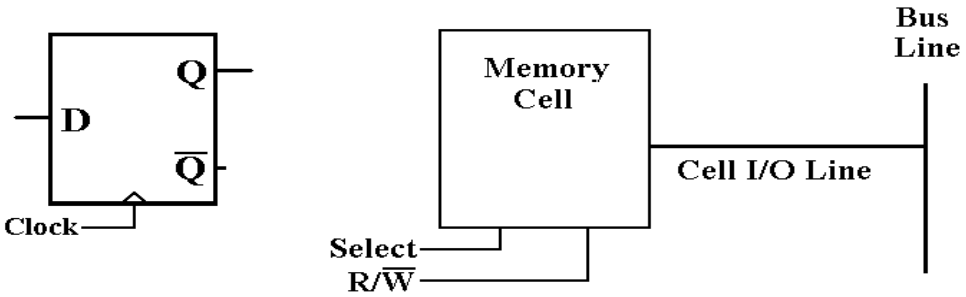
Bits	19		5	4		0
Contents	Tag			Offset in block		

22 A SDRAM memory unit is connected to a CPU via a 500 MHz memory bus.
If the memory bus is 128 bits wide and the SDRAM is operated at Double Data Rate (it is a DDR–SDRAM), what is the maximum burst transfer rate in bytes per second?

ANSWER: The bus is 128 bits (16 bytes) wide, so it delivers 16 bytes at a time.
It is DDR so it transfers 32 bytes per clock pulse.
It is DDR at 500 MHz, so it delivers $500 \bullet 10^6$ collections of bytes per second.
The data rate is $32 \bullet 500 \bullet 10^6$ bytes per second or $16{,}000 \bullet 10^6$ bytes per second.
This is $16 \bullet 10^9$ bytes per second, which is about 14.9 GB/sec.
Note: I shall accept 16.0 GB/sec, though it is technically incorrect.

23 You are given a D flip–flop, such as shown in the left diagram. You are to design a memory cell as shown in the right diagram. The memory cell I/O line must be connected directly to the bus line, and not pass through any other gates.

The memory cell is controlled by two inputs: Select and R/$\overline{W}$.
If Select = 0, the cell is inactive. If Select = 1, it is either being read or written to.
If R/$\overline{W}$ = 0, the cell is being written to, and changes state according to its D input.
If R/$\overline{W}$ = 1, the output of the cell (Q) is being placed on the bus line. $\overline{Q}$ is not used.



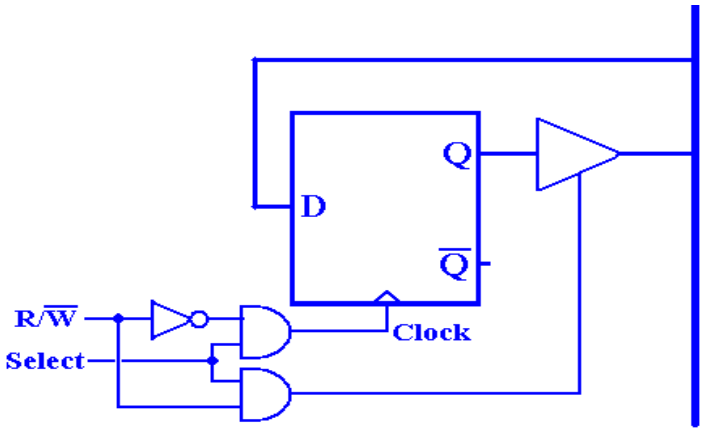
Design the circuit using only gates that will be internal to the Memory Cell. Recall that the D flip–flop control signal called "clock" is just a pulse that causes the flip–flop to change states.

ANSWER: Here we note that the control signals to the cell are as follows:

The cell is being written to if and only if Select = 1 and R/$\overline{W}$ = 0.
The cell is being read if and only if Select = 1 and R/$\overline{W}$ = 1.

The cell receives data only when its clock input goes high. Remember that this is an input to the flip–flop that activates it and enables it to receive data. The name is standard, but a bit misleading. While this might be connected to the system clock, it need not be.

Here is the circuit.



24 A computer has a 24–bit MAR. Each 16–bit word is individually addressable. All answers larger than 128 should be given as a power of two.

a) How many words can this computer address?
b) Assuming 8–bit bytes, how many bytes can this computer address?
c) What is the size of the MBR?
d) If this memory is implemented with 2MB (2^{21} by 8) chips, how many of these chips will be needed?
e) This memory is set up as **low–order interleaved**. Identify the address lines that go to each chip and the address lines used to select the memory bank.

ANSWERS: a) The MAR size is 24 bits; the computer can address **2^{24} words.**
This is also $2^4 \bullet 2^{20}$ words = $16 \bullet 2^{20}$ words = **16 M words**.
This has a 24–bit address: $A_{23} - A_0$.

b) One 16-bit word = 2 bytes, so the memory size is $2 \bullet 2^{24}$ bytes or **2^{25} bytes.**
This is also $2^5 \bullet 2^{20}$ bytes = $32 \bullet 2^{20}$ bytes = **32 M bytes**.

c) The MBR is **16 bits** in size, the same as the size of the addressable unit.

d) The chip count is given by dividing the memory size by the chip size. As the chips are sized in bytes, we use this as a common measure.
N = 2^{25} bytes/ 2^{21} bytes = 2^4 = **16**.

e) The memory chips are 2MB. We need memory banks of size 2 megawords, with the word size being 16 bits = 2 bytes. Each bank of memory will have two of the chips, one for bits 15 – 8 and the other for bits 7 – 0.

There will be eight banks of memory, each holding two memory chips. This accounts for the 16 chips.

Three bits will select the bank: **$A_2 - A_0$**.
The high–order twenty one bits (**$A_{23} - A_3$**) are sent to each bank, and hence to each chip in the bank.