1. (15 pts) **We're going to play Final (exam) Jeopardy!** Associate the following answers with the appropriate question. (You are given the "answers": Pick the "question" that goes best with each "answer".) The first two have been done for you.

**"Answers:"**

1 **(t)**. A lame attempt at humor.

2 **(u)** Anywhere but here.

3. That portion of the system bus used for the transfer of an Address.
4. A technique in which the occurance of one event on a bus follows and depends only on the occurance of a previous event.
5. To maximize the efficient use of an expensive system.
6. To make Memory appear to cost as much as the cheapest element and perform as well as the fastest element.
7. Technique used in CD-ROM Drives.
8. A technique in which an I/O device is periodically checked to see if it needs service.
9. A binary digit appended to a group of binary digits to make the sum of all the digits either always odd or always even.
10. A structure that holds recent mappings of virtual to physical addresses.
11. A structure that holds the contents of recently referenced memory locations.
12. An unscheduled subroutine call.

**"Questions:"**

a) What is synchronous timing?
b) What is asynchronous timing?
c) What is a Cache?
d) What is an Address Translation Lookaside Buffer?
e) What is the Address Bus?
f) What is the Data Bus?
g) What is a Parity error?
h) What is a Register?
i) What is DMA?
j) What is an interrupt?
k) What is a Page?
l) What is a Floating Point number?
m) What is a Page Fault?
n) What is a Parity bit?
o) What is a Sign bit?
p) What is the goal of the Memory Heirarchy?
q) What is polling?
r) What is the goal of multiprogramming?
s) What is Constant Linear Velocity?
t) What is this question?
u) Where would I rather be right now than where I am?

2. (2) What is the difference between memory access time and memory cycle time?

3. (4) Caches can be either Virtually Addressed or Physically Addressed. Explain the difference, and give an advantage and a disadvantage for each.

4. (2) What is the difference between volatile and non-volatile memory?

5. (4) Explain the principle of Locality and why it is important.

6. (0) Morning Class: "What... is your favourite colour?"

   (0) Afternoon Class: "What... is the air-speed velocity of an unladen swallow?"

7. (6) Given a logical 20-bit address and a 128K-byte physical memory for a byte-addressable machine,

   How big is the physical address space?

   How big is the virtual address space?

   Assuming 1K-byte pages, how many page frames are there? How many pages?

   Assuming 8K-byte pages, how many page frames are there? How many pages?

8. (3 pts) For the following 8-bit patterns, add a 9th bit to create:

   Even Parity: 00011011_

   Odd Parity: 00101101_

   What is the Hamming distance between these two codewords?

9. (8 pts) Here is a 12-bit Error Correction code format (same one used in class):

$$d_8\ d_7\ d_6\ d_5\ C_4\ d_4\ d_3\ d_2\ C_3\ d_1\ C_2\ C_1$$

   a. Given the *data* bit pattern

      **1 1 0 1 0 0 1 0**

   in a machine using the above ECC code, what bit pattern gets sent to memory? (No credit will be given without work being shown.)

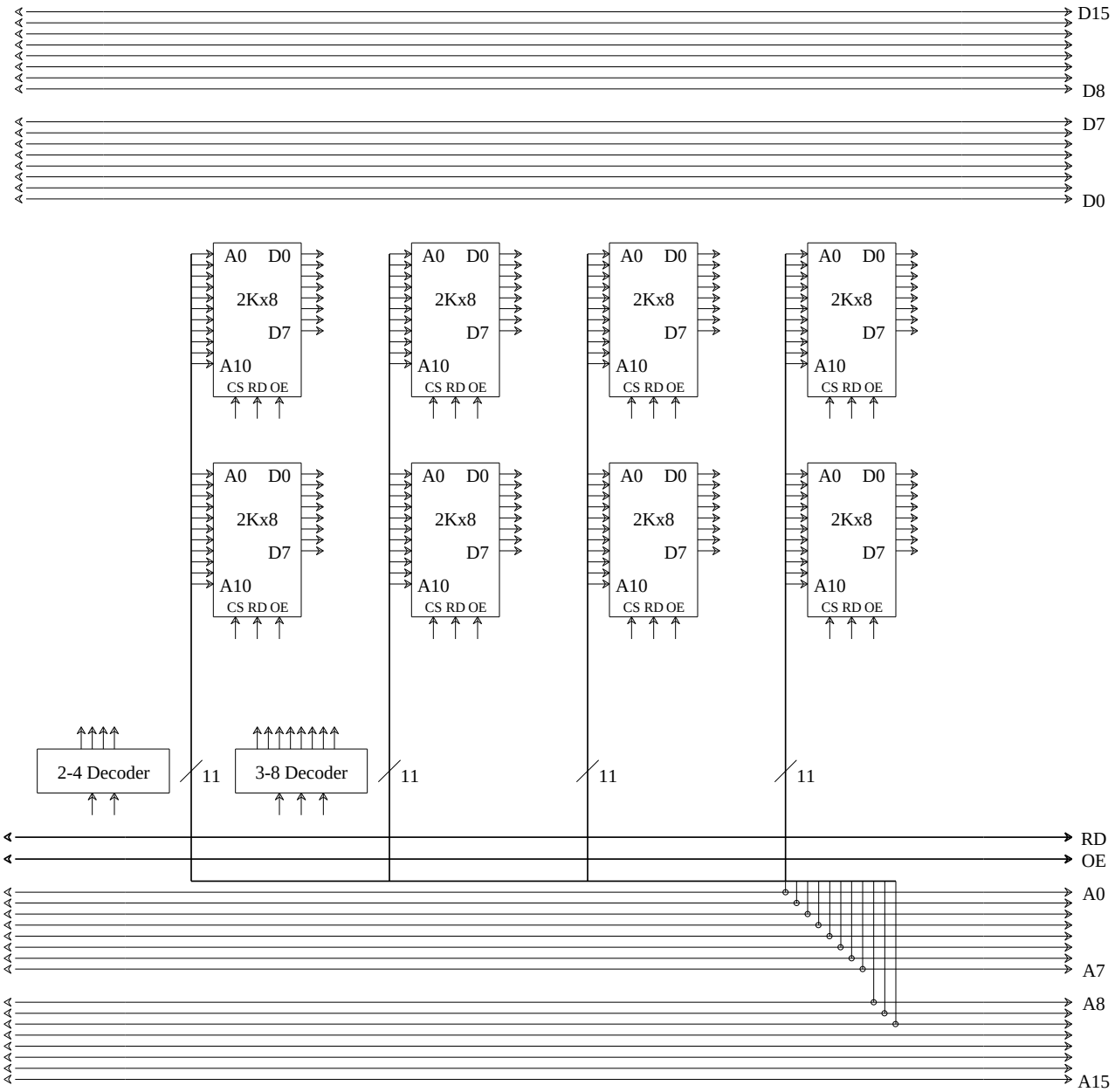
   b. In this same machine, the following bit pattern is retrieved from memory:

      **1 0 1 1 0 1 0 1 0 1 1 0**

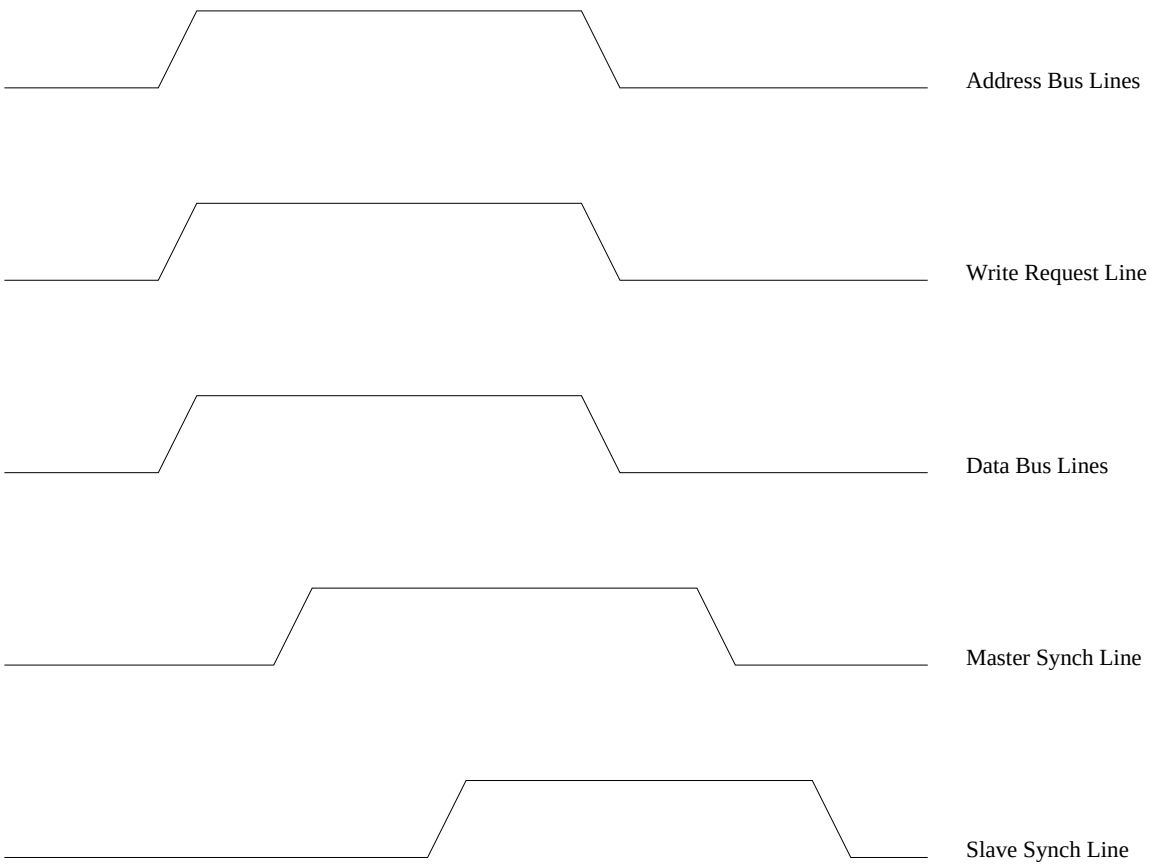
   Assuming the above Error Correction code format, identify and correct any errors that may have occurred during transmission or storage. (No credit will be given without work being shown.)

10. (7 pts) Add the connections to the following diagram necessary to create a 16Kx8 memory. Not all of the hardware shown is required to perform this task.

CS - Chip Select
OE - Output Enable
RD - Read (Read/Write, technically)



11. (5 pts) Add arrows to the following diagram to indicate which signal transitions **cause** other transitions to occur.




Now explain in words (briefly) what the arrows you have added are doing. (What is the sequence of events in words?)

12. (25) Megatron Corporation makes the "Megamax Infinity", a byte-addressable computer with a 16-bit word size. In this machine the bus to main memory is 32 bits wide, and accessing Main memory takes 8 clock cycles. In order to improve performance, they are considering adding a 64-byte Direct-Mapped cache with a line size of 2 words and an access time of 1 cycle. Given the following address reference sequence (in Hex):

**0xB4,0xB6,0xB8,0x38,0x13**

a) Write down how you are partitioning each address (which bits are the Tag, offset, etc.)

b) In the table below, fill in the proposed Cache's Tag values after each memory reference has been processed. If it is a hit, mark the entry number to indicate this, and if it is a miss enter what the new tag should be.

| **Tag Array** | Contents of Tag Array after processing address (Time -> ) | | | | | |
|---|---|---|---|---|---|---|
| Entry Number | Initial Contents | 0xB4 (10110100) | 0xB6 (10110110) | 0xB8 (10111000) | 0x38 (00111000) | 0x13 (00010011) |
| 0 | 10 | | | | | |
| 1 | 01 | | | | | |
| 2 | 11 | | | | | |
| 3 | 10 | | | | | |
| 4 | 00 | | | | | |
| 5 | 01 | | | | | |
| 6 | 10 | | | | | |
| 7 | 01 | | | | | |
| 8 | 10 | | | | | |
| 9 | 01 | | | | | |
| A | 11 | | | | | |
| B | 10 | | | | | |
| C | 00 | | | | | |
| D | 01 | | | | | |
| E | 00 | | | | | |
| F | 01 | | | | | |

What is the Average Memory access time for this sequence of references?

Now fill in the contents of the Data array after processing the given address references. Write down only the ones that change.

| **Data Array** | Data Array Contents after processing address | | |
|---|---|---|---|
| Entry Number | 0xB4 | 0xB6 | 0xB8 |
| 0 | | | |
| 1 | | | |
| 2 | | | |
| 3 | | | |
| 4 | | | |
| 5 | | | |
| 6 | | | |
| 7 | | | |
| 8 | | | |
| 9 | | | |
| A | | | |
| B | | | |
| C | | | |
| D | | | |
| E | | | |
| F | | | |

| Memory Contents at Hex Address **XY** | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Most Significant Digit (**X**) | Least Significant Digit (**Y**) | | | | | | | | | | | | | | | |
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
| 0 | 23 | 20 | 6d | 61 | 74 | 74 | 27 | 73 | 20 | 67 | 76 | 69 | 65 | 77 | 20 | 73 |
| 1 | 68 | 65 | 6c | 6c | 20 | 73 | 63 | 72 | 69 | 70 | 74 | 0a | 73 | 65 | 74 | 20 |
| 2 | 67 | 66 | 69 | 6c | 65 | 20 | 3d | 20 | 24 | 31 | 0a | 09 | 65 | 63 | 68 | 6f |
| 3 | 20 | 2d | 6e | 20 | 22 | 67 | 67 | 72 | 61 | 70 | 68 | 20 | 24 | 67 | 66 | 69 |
| 4 | 6c | 65 | 2e | 2e | 2e | 22 | 20 | 0a | 09 | 65 | 63 | 68 | 6f | 20 | 22 | 2e |
| 5 | 73 | 70 | 20 | 32 | 22 | 20 | 3e | 21 | 20 | 2f | 74 | 6d | 70 | 2f | 74 | 65 |
| 6 | 6d | 70 | 24 | 24 | 2e | 6e | 72 | 0a | 09 | 65 | 63 | 68 | 6f | 20 | 22 | 2e |
| 7 | 70 | 6f | 20 | 2b | 30 | 2e | 35 | 69 | 22 | 20 | 3e | 3e | 20 | 2f | 74 | 6d |
| 8 | 23 | 7b | 92 | 08 | 22 | 41 | 85 | 32 | 69 | 73 | 11 | 35 | 97 | 54 | 31 | 48 |
| 9 | 88 | 73 | 48 | 72 | 98 | 21 | 42 | 85 | 62 | 65 | 90 | 84 | 31 | 56 | 55 | 83 |
| a | 43 | 64 | 84 | 36 | 59 | 3c | 8a | 95 | 3b | 8f | 0e | 41 | 7a | 40 | 2b | 3c |
| b | 4c | d4 | c7 | 82 | a0 | 38 | f9 | c6 | 29 | a3 | d0 | 9c | 7d | 41 | 2b | 75 |
| c | 54 | 69 | 9c | 3b | b0 | 2a | d9 | 3e | 45 | 72 | 6e | f0 | f9 | 3f | a0 | 0a |
| d | 60 | 89 | 43 | d8 | c0 | e7 | 49 | 76 | 59 | 21 | 2c | c8 | a8 | f2 | 87 | 43 |
| e | 76 | 8f | 2e | a9 | ff | 38 | ae | 65 | dd | cf | 21 | 84 | ce | e4 | 34 | 51 |
| f | 8a | 65 | 30 | 2f | c9 | 3a | 58 | 72 | 3e | a0 | 4f | 38 | 96 | 47 | 21 | 80 |

13. (22) Megatron decided to experiment with using a 40-byte 5-way Set Associative Cache (instead of the Direct-mapped Cache). Remember, the Megamax Infinity is a byte-addressable machine with a 16-bit word size, a line size of 2 words, a 32-bit bus to main memory, and a Main Memory access time of 8 cycles. The Cache access time is still 1 cycle. Given the same address reference sequence (in Hex) as before:

**0xB4,0xB6,0xB8,0x38,0x13**

a) Write down how you are partitioning each address (which bits are the Tag, offset, etc.)

b) In the table below, fill in the proposed Cache's Tag values after each memory reference has been processed. If it is a hit, mark the entry number to indicate this, and if it is a miss enter what the new tag should be. Use an LRU replacement scheme.

| **Tag Array** | | | Contents of Tag Array after processing address (Time -> ) | | | | | |
|---|---|---|---|---|---|---|---|---|
| Set Number | Entry Number | Age | Initial Contents | 0xB4 (10110100) | 0xB6 (10110110) | 0xB8 (10111000) | 0x38 (00111000) | 0x13 (00010011) |
| 0 | 0 | newest-1 | 0 0 0 1 0 | | | | | |
| | 1 | oldest | 0 0 1 1 1 | | | | | |
| | 2 | newest-3 | 1 0 0 1 1 | | | | | |
| | 3 | newest | 1 0 1 1 1 | | | | | |
| | 4 | newest-2 | 1 0 0 1 0 | | | | | |
| 1 | 0 | oldest | 0 0 0 1 0 | | | | | |
| | 1 | newest-3 | 0 1 0 0 1 | | | | | |
| | 2 | newest | 0 1 0 1 1 | | | | | |
| | 3 | newest-1 | 1 1 1 1 0 | | | | | |
| | 4 | newest-2 | 1 0 1 1 1 | | | | | |

What is the Average Memory access time for this sequence of references?

14. (12 pts) The following tables contain some of the information about a segmented, paged virtual memory system and certain select memory locations. Total physical memory size is 4K bytes, and the page size is 512 bytes. All numbers in this table are in Hex unless otherwise noted.

**Segment Table**

| Entry Number | Presence Bit | Page Table |
|---|---|---|
| 0 | 1 | 3 |
| 1 | 0 | 0 |
| 2 | 1 | 1 |
| 3 | 1 | 7 |
| 4 | 1 | 2 |
| 5 | 1 | 2 |
| 6 | 1 | 2 |
| 7 | 1 | 2 |

**Page Table 0**

| Entry Number | Present? (1=Yes) | Disk Addr | Frame Number |
|---|---|---|---|
| 0 | 1 | 1234123 | 0x4 |
| 1 | 0 | 0893748 | 0x7 |
| 2 | 1 | 2489567 | 0x1 |
| 3 | 1 | 9623873 | 0x5 |
| 4 | 1 | AE76BD3 | 0x2 |
| 5 | 0 | 328759A | 0x3 |
| 6 | 1 | 56D87AC | 0x0 |
| 7 | 1 | 10A876D | 0x6 |

**Page Table 2**

| Entry Number | Present? (1=Yes) | Disk Addr | Frame Number |
|---|---|---|---|
| 0 | 1 | 1234123 | 0x1 |
| 1 | 0 | 0893748 | 0x3 |
| 2 | 1 | 2489567 | 0x5 |
| 3 | 1 | 9623873 | 0x7 |
| 4 | 1 | AE76BD3 | 0x1 |
| 5 | 0 | 328759A | 0x2 |
| 6 | 1 | 56D87AC | 0x4 |
| 7 | 1 | 10A876D | 0x6 |

**Memory**

| Address | Contents |
|---|---|
| 0x00A4 | 0x76 |
| 0x01A4 | 0x73 |
| 0x02A4 | 0x32 |
| 0x03A4 | 0x46 |
| 0x04A4 | 0x30 |
| 0x0AA4 | 0x29 |
| 0x05A4 | 0xa9 |
| 0x09A4 | 0x74 |
| 0x0BA4 | 0x05 |
| 0x0CA4 | 0x23 |
| 0x0DA4 | 0xE3 |
| 0x07A4 | 0xAE |
| 0x0EA4 | 0x92 |

**Page Table 5**

| Entry Number | Present? (1=Yes) | Disk Addr | Frame Number |
|---|---|---|---|
| 0 | 1 | 1234123 | 0x2 |
| 1 | 0 | 0893748 | 0x3 |
| 2 | 0 | 2489567 | 0x4 |
| 3 | 1 | 9623873 | 0x4 |
| 4 | 1 | AE76BD3 | 0x6 |
| 5 | 0 | 328759A | 0x7 |
| 6 | 1 | 56D87AC | 0x0 |
| 7 | 1 | 10A876D | 0x6 |

**Page Table 7**

| Entry Number | Present? (1=Yes) | Disk Addr | Frame Number |
|---|---|---|---|
| 0 | 1 | 1234123 | 0x5 |
| 1 | 0 | 0893748 | 0x6 |
| 2 | 1 | 2489567 | 0x1 |
| 3 | 1 | 9623873 | 0x2 |
| 4 | 1 | AE76BD3 | 0x4 |
| 5 | 0 | 328759A | 0x2 |
| 6 | 1 | 56D87AC | 0x5 |
| 7 | 1 | 10A876D | 0x6 |

For each of the following convert the virtual address into a physical address (if possible) and write down the value of the memory location corresponding to the address. If it is not possible to do so, explain why.

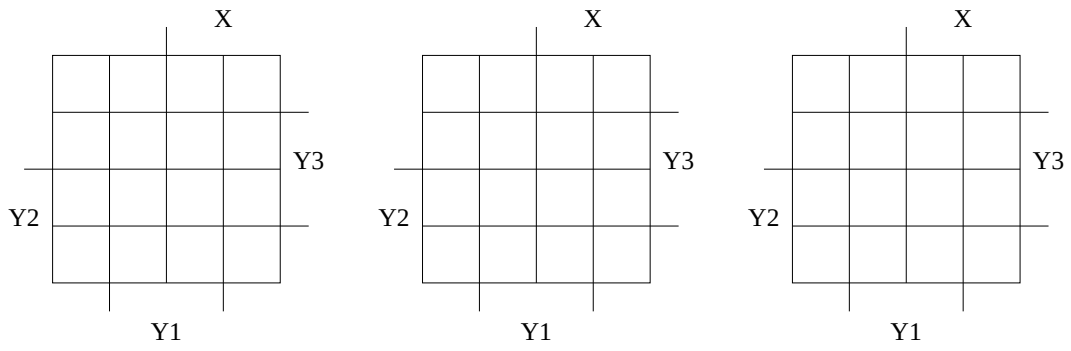
**0x30A4** (**0 1 1 0 0 0 0 1 0 1 0 0 1 0 0** in binary).

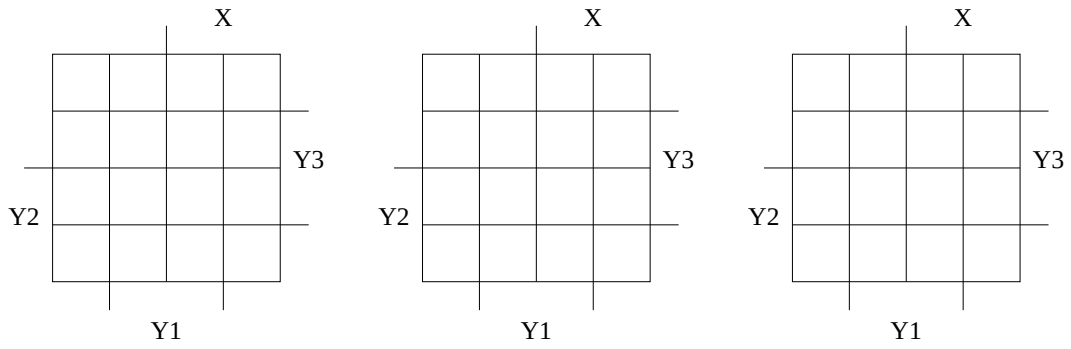
**0x1EA4** (**0 0 1 1 1 1 0 1 0 1 0 0 1 0 0** in binary).

**0x76A4** (**1 1 1 0 1 1 0 1 0 1 0 0 1 0 0** in binary).

15. (20) Given the following table, draw the Karnaugh maps for Y1', Y2', and Y3' and Z in terms of X, Y1, Y2 and Y3, and then write **minimum** boolean equations for each.

| Present State (Y1 Y2 Y3) | Next State X=0 (Y1' Y2' Y3') | Next State X=1 (Y1' Y2' Y3') | Output X=0 | Output X=1 |
|---|---|---|---|---|
| 000 | 101 | 111 | 0 | 0 |
| 001 | 101 | 111 | 1 | 0 |
| 010 | 101 | 101 | 0 | 1 |
| 011 | 111 | 110 | 1 | 1 |
| 100 | 100 | 110 | 0 | 0 |
| 110 | 000 | 010 | 0 | 1 |
| 111 | 111 | 110 | 1 | 1 |



16. (15 pts) Given the following Karnaugh maps, implement the sequential machine using an T FF for Y1, a SR FF for Y2, and a JK FF for Y3. You do not need to draw the gates, but you do need to write down the **minimized** input equations for each of the inputs of each of the Flip Flops in the circuit.

**Y1'**

| Y2 \ Y3 | X=0 | X=1 |
|---|---|---|
| 0 | 1 | 1 |
| 1 | 1 | d |
| 3 | 1 | d |
| 2 | 1 | 1 |

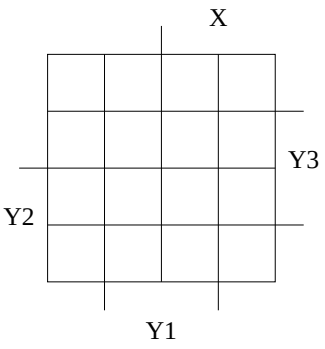
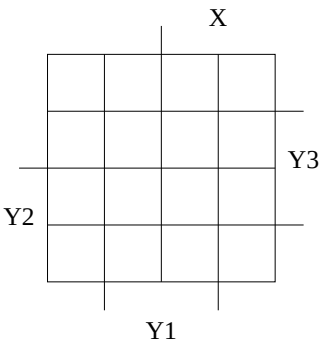
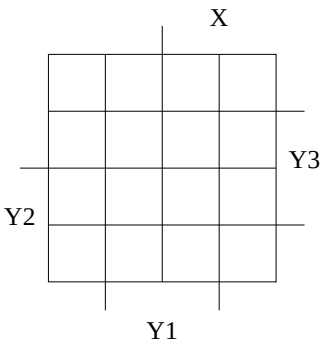
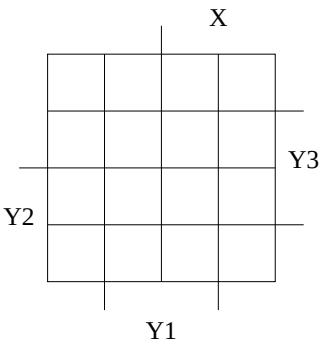
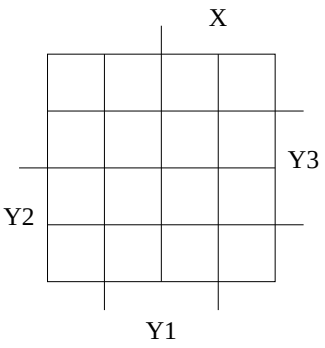
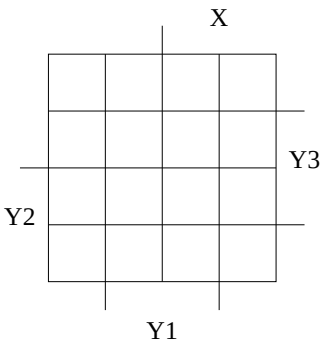
Y1

**Y2'**

| Y2 \ Y3 | X=0 | X=1 |
|---|---|---|
| 0 | d | |
| 1 | 1 | d |
| 3 | | d |
| 2 | 1 | 1 |

Y1

**Y3'**

| Y2 \ Y3 | X=0 | X=1 |
|---|---|---|
| 0 | 1 | 1 |
| 1 | 1 | 1 |
| 3 | 1 | 1 |
| 2 | 1 | 1 |

Y1