In the real midterm exam,

You may:	**You must not:**
Open your textbook and your notebook. Refer to eTextbook and your memo in cloud. Refer to any CSS422 Canvas materials including this rehearsal and key answers. Use a calculator (no number conversion)	Use any Internet search engine. Use number-converting software tools such as: https://www.rapidtables.com/convert/number/decimal-to-hex.html https://www.exploringbinary.com/floating-point-converter/ Use ARM encoding/decoding software tools: https://armconverter.com/ Use Keil uVersion, VisUAL, or LogiSim.

CSS422 Final Exam Rehearsal

Q1. Combinational Circuits

Q1-1. Expand and simplify (~A + ~B)(A + B)
Don't use the K-map.

Q1-2. Convert (A + B + ~C)(A + ~B + ~C)(A + ~B + C) into a simplified product of sums

Q1-2-i: First rewrite it as ~F. Then, apply de Morgan's law to simplify ~F into a sum of products.

Q1-2-ii: Fill the following K-map of ~F

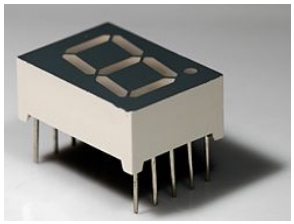
BC / A	00	01	11	10
0				
1				

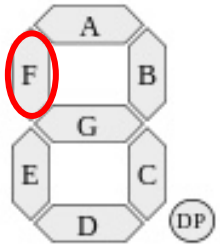
What is the simplified ~F?

Q1-2-iii: Negate ~F back to F and use de Morgan's law to make it back to a product of sums.

Q1-3. Consider the following 7-segment display. Design a combinational circuit that satisfies the following specification:

- Receive four inputs (P, Q, R, S) where P is MSB and S is LSB.
- Assumes that PQRS represents a binary number 0000 to 1010, thus ignoring higher number including 1011, 1100, 1101, 1110, and 1111.
- Set the output to turn on segment F of this display.

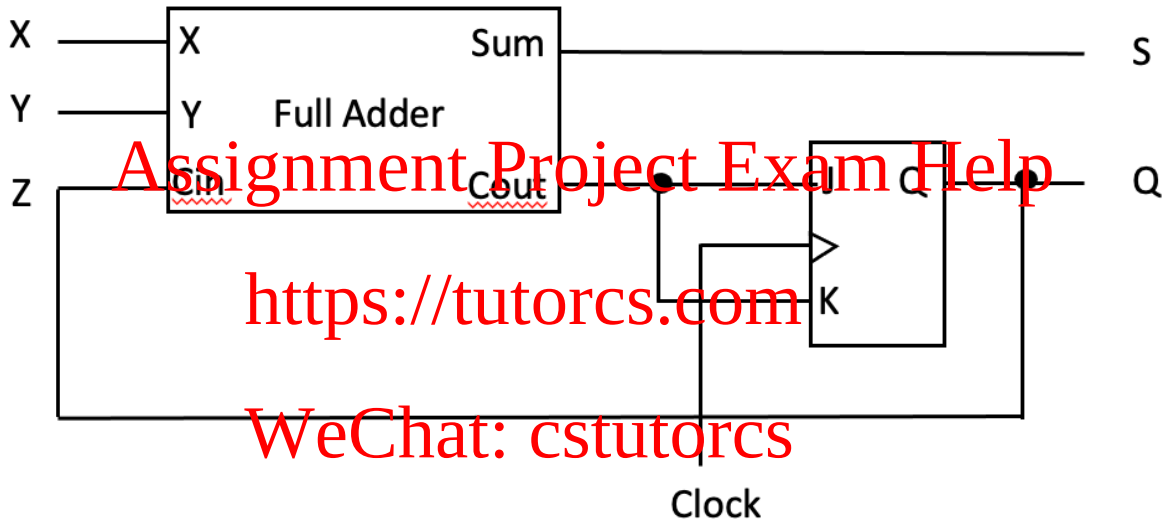




rs / pq	00	01	11	10
00				
01				
11				
10				

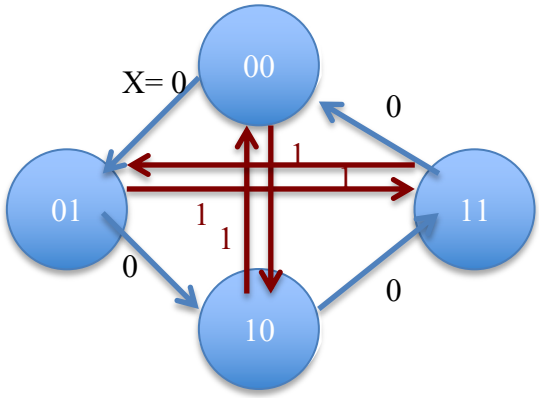
Q2. Sequential Circuits

Q2-1. A sequential circuit has one flip-flop; two inputs, X and Y; and one output, S. It consists of a full-adder circuit connected to a JK flip-flop, as shown. Fill in the truth table for this sequential circuit by completing the Next State and Output columns.



Present State Q(t)	Inputs X Y	Cout = J = K	Sum	Next State Q(t + 1)	Output S
0	0 0				
0	0 1				
0	1 0				
0	1 1				
1	0 0				
1	0 1				
1	1 0				
1	1 1				

Q2-2. Design the following sequential circuit.



	X = 0	X = 1
Currenet State	Next State	Next State
00		
01		
10		
11		

	X = 0	X = 1	X = 0		X = 1		X = 0		X = 1	
curr	next	next	JK FF -A				JK FF - B			
AB	AB	AB	Ja	Ka	Ja	Ka	Jb	Kb	Jb	Kb
00										
01										
10										
11										

Ja =

X \ AB	00	01	11	10
0				
1				

Ka =

X \ AB	00	01	11	10
0				
1				

Jb =

X \ AB	00	01	11	10
0				
1				

Kb=

X \ AB	00	01	11	10
0				
1				

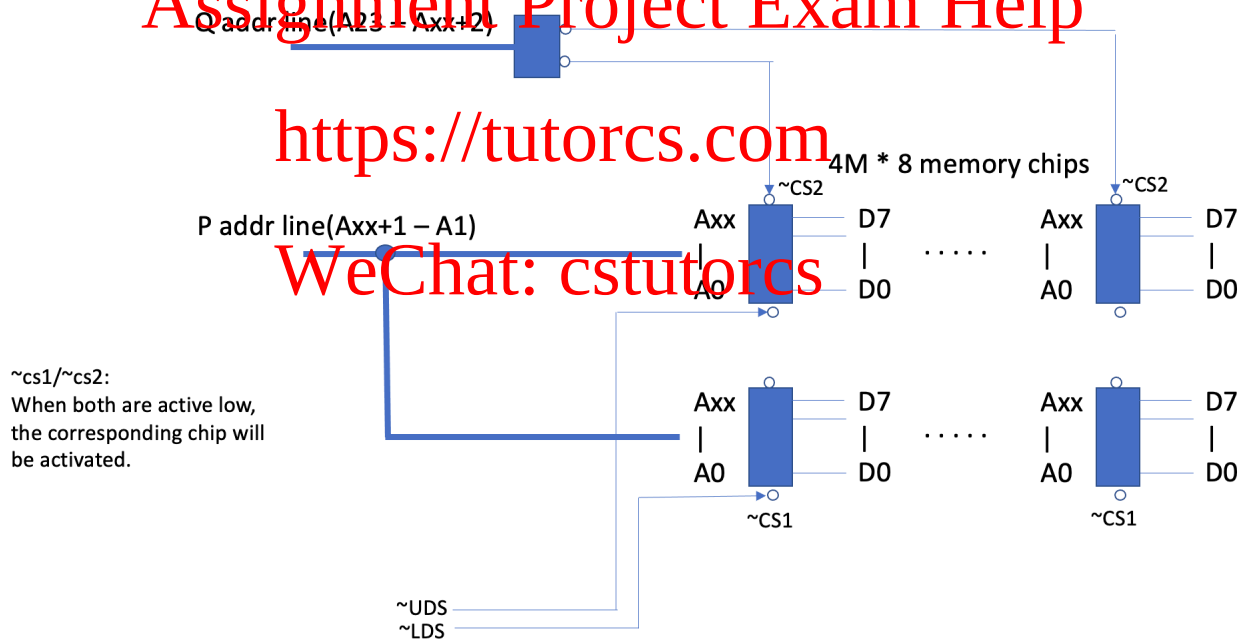
Q3 Memory and I/O

Q3-1. Motorola 68K has 16 data lines: D15 – D0; 23 address lines: A23 – A1; and ~UDS and ~LDS data strobe pins. Each of ~UDS and ~LDS is low-active to choose D15 – D8 and D7 – D0 respectively. For example, to access 1 byte at 0x000008, 68K sets A23 – A4 in 0, whereas A3 = 1, A2 = 0, A1 = 0, ~UDS = 1, and ~LDS = 0. Assume that we want to connect 68K to a memory system M that is built with a number of CY7C1079DV33 memory chips, each with 4M x 8-bit wide data. Answer the following four questions.

Q3-1-i. How many bytes of memory should the memory system M have (The capacity of the memory system in bytes)?

Your answer:

Q3-1-ii. How many address lines does this memory chip, (CY7C1079DV33) has? In the diagram below, you'll compute P (Thus Axx in the diagram should be A_{P-1}).



Your answer:

Q3-1-iii. How many selection bits of each 2 memory chips must be used? In other words, how many upper address lines of 68K should be used for choosing a group of 2 memory chips? In the above diagram, you'll compute Q.

Your answer

Q3-1-iv. How many memory chips in total is needed to construct this memory system M?

Your answer

Q4. CISC/RISC and Cache Memory

Q4-1. Assume that Intel i486 processor has an L1 cache with the following specifications:

- 32-bit wide address and 32-bit wide data busses
- On-chip instruction cache
- Cache is 8K bytes, organized as a 4-way set associative
- Cache line (block) size = 16 bytes
- 100 MHz clock frequency
- Average cache hit rate = 90%
- Instructions located in cache execute in 1 clock cycle

Instructions that are not found in the on-chip cache will cause the processor to stop all program execution and do a ***burst memory read access*** of one refill line from main memory to the cache. For this memory design, burst accesses from main memory requires an address set-up time of 32 clock cycles, and then all subsequent burst fetches from main memory require 2 clock cycles per memory fetch.

Q4-1-i. Calculate 1 clock cycle, (i.e., the time length of 1 clock)

Your answer:

Q4-1-ii. How many memory fetches must be done upon a cache miss?

Your answer:

Q4-1-iii. Calculate CPU cycles required to read data from system memory upon a cache miss. (# clocks but not time)

Your answer:

Q4-1-iv. What is the effective instruction execution time for this i486 processor?

Your answer:

Q4-2. Review Homework8's Q1 – Q3.

Q4-3.i Consider a byte-addressable memory space with 10 address lines, (i.e., 1KB size). A CPU has a direct-mapped cache with a 128B capacity and its cache line size is 8 bytes.

Which of the following for-loop can utilize direct-mapping cache better? Loop A or Loop B?


```
// Loop A
int c[256];
for ( int i = 0; i < 255; i++ )
   a[i] += a[i+1];

// Loop B
int c[256];
for ( int i = 0; i < 224; i++ )
   a[i] += a[i+32];
```


Your answer:

Q4-3-ii. Let's assume that two CPUs, (say CPUs A and B) execute the following two code snippets (code A and B), respectively. Also assume that "int array[N]" is shared among these two CPUs.

```
// Code A to be executed by CPU A
for ( int i = 0; i < N/2; i++ )
   array[i] = 0;

// Code B to be executed by CPU B
for ( int i = N / 2; i < N; i++ )
   array[i] = 0;
```

Do not worry about any CPU synchronization (which you'll study about in CSS430 OS). Just focus on cache memory. Which policy of "write invalidation" or "write update" would work better or faster?

Your answer: