

Midterm Computer Organisation CSE1400

Please read the following information carefully!

- This exam consists of 16 multiple-choice questions. Each question is worth 250 points.
- You have 90 minutes to complete this exam.
- Before you hand in your answers, check that your multiple-choice form contains your name and student number, also filled in using the boxes.
- Opening this exam before you are instructed to start is **strictly prohibited**.
- The use of the book, notes, calculators, smart watches, and other aids is **strictly prohibited**.
- Note that the order of the letters next to the boxes on your multiple-choice sheet may **not always be A-B-C-D!**
- Fill in the answer form with **(dark) pencil** or **pen**. If you make a mistake on the answer form, you need to either erase the mistake or copy all answers to a new form.

1. History has shown that capitalism almost came to a grinding halt with respect to claiming the invention of the “digital computer”. The patent was only awarded in 1973, when digital computers already entered their second generation.

The lucky winner(s) of the IP lawsuit was/were:

- A. John Atanasoff
- B. Mauchly and Eckert
- C. Howard Aiken
- D. Charles Babbage

2. In the past, the Netherlands could see some cold winters. In some cases snow would fall for days and temperatures would stay below 0 for weeks. These extreme conditions could cause a school to close prematurely or not even open at all. In the Netherlands, schools only have “ijsvrij” (a snow day) occur when certain conditions are hit.

The current temperature is stored in °C as T . T is represented as a 3 bit unsigned number $T = t_2t_1t_0$ where t_2 is the most significant bit; temperatures that are higher than 7 °C are stored as 111, temperatures that are below zero are stored as 000. On any particular day, students will have “ijsvrij” when temperature T drops below 3.

Find a minimal Sum-of-Products equation that is only true when students have “ijsvrij”.

- A. $\overline{t_2} \cdot t_1 + \overline{t_2} \cdot \overline{t_0}$
- B. $\overline{t_2} \cdot \overline{t_1} + \overline{t_1} \cdot \overline{t_0}$
- C. $\overline{t_2} \cdot \overline{t_1} + \overline{t_2} \cdot \overline{t_0}$
- D. $\overline{t_2}$

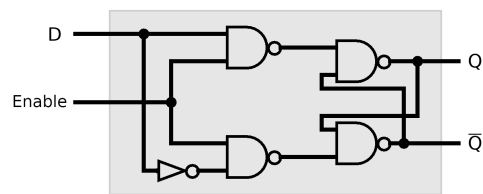
Solution: We first write down the truth table for T and function f .

t_2	t_1	t_0	f
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

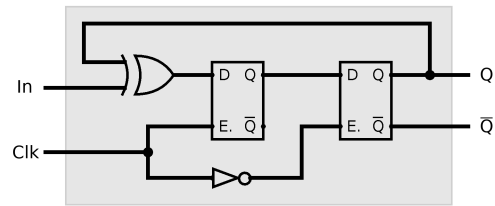
Next, we write down the Karnaugh map for this truth table to find the sum-of-products form.

$t_2 \backslash t_1 t_0$	00	01	11	10
0	1	1	0	1
1	0	0	0	0

We see that for $f = 1$, $t_2 = 0$ and $t_1 = 0$ or $t_0 = 0$. In minimal sum-of-products form: $f = \overline{t_2} \cdot \overline{t_1} + \overline{t_2} \cdot \overline{t_0}$.



(a) A gated D-latch.



(b) Stephen's flip-flop circuit, using two D-latches from Figure 1a.

Figure 1

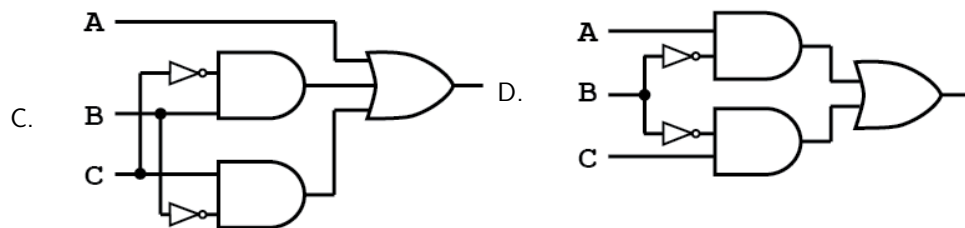
3. Stephen has built a flip-flop circuit using two gated D-latches, an XOR gate and a NOT gate, as shown in Figure 1b. Indicate what type of flip-flop Stephen has created, and whether it is positive (leading) edge or negative (trailing) edge triggered.

- A. Positive edge triggered master-slave D flip-flop
- B. Negative edge triggered master-slave D flip-flop
- C. Positive edge triggered T flip-flop
- D. Negative edge triggered T flip-flop**

Solution: This looks like the negative edge triggered master-slave flip-flop from Figure A.27a in the book, except that the output of the whole circuit is looped back to the input side and combined with the "In" signal using an XOR gate. This has the following effects:

- When the input is 0, the input to the first D-latch will be whatever used to be at the output.
- When the input is 1, the input to the first D-latch will be the inverse of the output.

In other words, this flip-flop has toggling behaviour, the same as a T flip-flop.



Solution: Applying DeMorgan's law a couple of times, we get:

$$f(A, B, C) = \overline{(A + \overline{B}) \cdot (\overline{A} + B \cdot (C + \overline{C}))}$$

$$f(A, B, C) = \overline{(A + \overline{B}) + (\overline{A} + B \cdot (C + \overline{C}))}$$

$$f(A, B, C) = (\overline{A} \cdot B) + (A \cdot \overline{B} \cdot (C + \overline{C}))$$

$$f(A, B, C) = \overline{A} \cdot B + A \cdot (\overline{B} + (\overline{C} + C))$$






$$f(A, B, C) = \overline{A} \cdot B + A \cdot (\overline{B} + (\overline{C} \cdot C))$$

$$f(A, B, C) = \overline{A} \cdot B + A \cdot \overline{B} \quad (\text{because } \overline{C} \cdot C = 0)$$

For answer A, the OR-gate using the C 's as input will always generate a 1 and therefore the AND-gate that follows it has no function. When discarding these two gates, the circuit from answer A directly represents the simplified formula.

6. The ducks and sharks rally that began at the Reasoning & Logic multiple-choice test is still going strong. With renewed slogans like "Dear sharks, apologies for our earlier remarks!" they still hope to re-establish peace between the factions.

Unfortunately their rally has also made them walk straight over a Karnaugh map. All we know is that the map represents a function with 3 (min)terms (T) and 5 variables (V). Which of the following is possible given these ducks and sharks?

		DC			
		00	01	11	10
BA	00	1		1	0
	01		1	1	
	11		1	1	1
	10	0	d	1	

A. All the ducks stepped on “don’t cares”, and the sharks stepped on zeroes.

B. All the ducks stepped on ones, and the sharks stepped on ones.

C. All the ducks stepped on zeroes, and the sharks stepped on “don’t cares”.

D. All the ducks stepped on “don’t cares”, and the sharks stepped on ones.

Solution:

A.

		<i>DC</i>			
		00	01	11	10
<i>BA</i>	00	1	d	1	0
	01	d	1	1	0
	11	0	1	1	1
	10	0	d	1	d

This has exactly 3 terms, with a total of 5 variables.

B.

		<i>DC</i>			
		00	01	11	10
<i>BA</i>	00	1	1	1	0
	01	1	1	1	1
	11	1	1	1	1
	10	0	d	1	1

This has exactly 4 terms, with a total of 6 variables.

C.

		DC			
		00	01	11	10
BA	00	1	0	1	0
	01	0	1	1	d
	11	d	1	1	1
	10	0	d	1	0

This has exactly 4 terms, with a total of 9 variables.

D.

		DC			
		00	01	11	10
BA	00	1	d	1	0
	01	d	1	1	1
	11	1	1	1	1
	10	0	d	1	d

This has exactly 3 terms, with a total of 4 variables.

7. You are following a course with Professor Alex as teacher. In his course, you can score points towards your final grade in all kinds of ways. So far, you have obtained 11100011_2 points from doing quizzes, AAD_{16} points worth of lab exercises, and 3241_5 points at the exam. Alex says that you can earn even more points if you can correctly calculate your final score! What is your total number of points earned so far?

- A. 3331_{10}
- B. 3406_{10}**
- C. 3630_{10}
- D. 3679_{10}

Solution:

A. If you would read 3241_5 as $3 \cdot 100 + 2 \cdot 25 + 4 \cdot 5 + 1 = 371$ (the 100 should be 125), you end up with the answer 3331_{10} .

B. The grand total is indeed 3406_{10} points:

$$11100011_2 = 227_{10}$$

$$AAD_{16} = 2733_{10}$$

$$3241_5 = 446_{10}$$

C. If you would read 11100011_2 as $111000011_2 = 451$, you end up with answer 3630_{10} .

D. If you would read AAD_{16} as $BBE_{16} = 3006$ (i.e. think that $D_{16} = 14_{10}$ instead of 13_{10} , and similarly for the B), you end up with answer 3679_{10} .

8. Prof. dr. Hook is tasked with improving the new floating point standard created by the CSE1400 team last year to be used for storing grades efficiently. The improvements should make sure that a grade between 1 and 10 (inclusive) can be represented exactly to a quarter grade point, for example: 9.75, 8.25, 5.75.

How many bits should he pick for the sign bit, mantissa, and exponent? Assume the exponent is represented as an unsigned integer.

A. 1 sign bit, 4 mantissa bits, 2 exponent bits

B. no sign bit, 5 mantissa bits, 2 exponent bits

C. no sign bit, 4 mantissa bits, 2 exponent bits

D. 1 sign bit, 4 mantissa bits, 3 exponent bits

Solution: To represent numbers with a quarter of a unit precision exactly, we need to be able to represent 2^{-2} . Additionally, we will need to be able to go up to 9.75 with the same precision. 9.75 can be represented exactly as $(1001.11)_2$. To write this number as an IEEE-754 number, we need to shift the decimal point 3 places to get $(1.00111)_2 \cdot 2^3$. This part after the decimal point shows that a mantissa of at least 5 bits is required. The exponent has to be at most 3, so 2 bits will be enough. No sign bit is required as grades are always a positive number.

9. When using a digital circuit that adds two four-bit 2's complement numbers ($a_3a_2a_1a_0$ and $b_3b_2b_1b_0$) that produces a four-bit sum ($s_3s_2s_1s_0$), it is possible that a positive or negative overflow occurs when adding together two numbers that are very large.

If you were to calculate whether a positive or negative overflow has occurred based on the input and output signals, which formulas should you use to do so?

- A. Positive overflow: $a_3 \cdot b_3 \cdot s_3$; Negative overflow: $\overline{a_3} \cdot \overline{b_3} \cdot \overline{s_3}$
- B. Positive overflow: $\overline{a_3} \cdot b_3 \cdot s_3$; Negative overflow: $a_3 \cdot \overline{b_3} \cdot s_3$
- C. Positive overflow: $a_3 \cdot b_3 \cdot \overline{s_3}$; Negative overflow: $\overline{a_3} \cdot \overline{b_3} \cdot s_3$
- D. Positive overflow: $\overline{a_3} \cdot \overline{b_3} \cdot s_3$; Negative overflow: $a_3 \cdot b_3 \cdot \overline{s_3}$**

Solution:

Two positive numbers both have 0 as MSB. If adding them yields a 1 as MSB, there is overflow. Thus, $\overline{a_3} \cdot \overline{b_3} \cdot s_3$ correctly detects overflow.

Two negative numbers both have 1 as MSB. If adding them yields a 0 as MSB, there is underflow. Thus, $a_3 \cdot b_3 \cdot \overline{s_3}$ correctly detects underflow.

10. Sonic the Porcupine is in the process of building a clock-based electronic circuit. Sonic being Sonic, he wants the circuit to work as fast as possible. Sonic knows that the speed of his circuit depends on the maximum number of logic gates that a signal needs to propagate through, so he has already optimised his circuit in such a way that the longest signal propagation path through his circuit is at most 10 gates between the input and the output wires.

With every rising edge of the clock signal, the state of the gates will change. During the time that the signal propagates through the gates, the output signal will be unstable. Sonic knows that the output signal cannot be stable all the time, but he does want the output of the circuit to be stable for at least 95 % of the time.

The propagation delay of each gate is 0.05 ns and the transition time of each gate is negligible. What is the maximum clock frequency that Sonic's circuit can operate with?

- A. 40 MHz
- B. 100 MHz**
- C. 400 MHz
- D. 1 GHz

Solution: The circuit needs $0.05 \cdot 10 = 0.5$ ns to stabilise every clock pulse. The circuit is allowed to stabilise 5 % of the time, so the time between two clock pulses can be $0.5/0.05 = 10$ ns, or 10^{-8} seconds. This corresponds with a clock frequency of 10^8 Hz, or 100 MHz.

11. Mr. Johnson has a digital counter clock on his desk that increments its displayed value by one every hour. However, he would like to use this counter instead to count down to an important event involving a country leaving a specific group of countries.¹

¹Any resemblance to actual persons, living or dead, or actual events is purely coincidental.

The clock's current value is 0001 0011 0011 0111 (BCD) and the important event will occur at hour 0001 1000 1001 0100 (Excess-2048) if the clock would just continue counting.

With what value should the countdown clock start when it is programmed to count down to the event (in 2's complement)?

- A. 0000 0101 0101 1101
- B. 0000 1011 0101 1011**
- C. 0001 0101 1100 1101
- D. 0001 1011 0101 1011

Solution:

- A. If you just subtract the two binary values from each other as if they were both 2C, the answer becomes $1373_{10} = 0000\ 0101\ 0101\ 1101_2$

B. Correct because:

$$\begin{aligned} 0001\ 0011\ 0011\ 0111_{BCD} &= 1337_{10} \\ 0001\ 1000\ 1001\ 0100_{E-2048} &= 4244_{10} \\ 4244 - 1337 &= 2907 \\ &= 0000\ 1011\ 0101\ 1011_{2C} \end{aligned}$$

- C. If you add the two values together instead of subtract them, you get $5581_{10} = 0001\ 0101\ 1100\ 1101_2$

- D. If you normalize the Excess-2048 by adding 2048 instead of subtracting 2048, the answer becomes $0001\ 1011\ 0101\ 1011_2$

12. Hannah commutes to EEMCS using public transport every day. She wants to make sure that she is following the shortest route, because less travel time means more time to study. To do so, she wants to find out all travel times in seconds of all possible routes. She knows that there are no reasonable routes through the Netherlands that take more than 10000 seconds. Hannah figured she would have to use the least number of bits possible to store each travel time.

Which number representation should she choose?

- A. Binary Coded Decimal
- B. Excess-10000
- C. Two's complement
- D. Unsigned integer**

Solution:

- A. Binary coded decimal could be used, but is inefficient as every decimal digit will take up 4 bits.
- B. Excess-10000 might sound reasonable, but will cover a range of -10000 to ∞ numbers depending on the number of bits used.
- C. Two's complement represents negative integers as well as positive, but we do not need negative integers for the number of seconds.
- D. Unsigned integer is the correct answer.**

13. Which of the following statements is **true** about an IEEE-754 32-bit number?

- A. $1/0$ is not representable as an IEEE-754 number.
- B. $2^{64} \cdot 2^{-64}$ is not representable as an IEEE-754 number.
- C. $2^{64} \cdot 2^{64}$ is not representable as an IEEE-754 number.**
- D. $2^{100} + 2^{100}$ is not representable as an IEEE-754 number.

Solution:

- A. The result of $1/0$ is positive infinity, which is representable in IEEE-754.
- B. Multiplying two numbers with these exponents will not cause an overflow or underflow as the resulting exponent is not larger than 127 or smaller than -126 (in fact, the exponent will be 0).
- C. This causes an overflow as the represented exponent range is $(-126, 127)$ and the total exponent will be 128.**
- D. Adding together two times 2^{100} will result in 2^{101} , this will still fit.

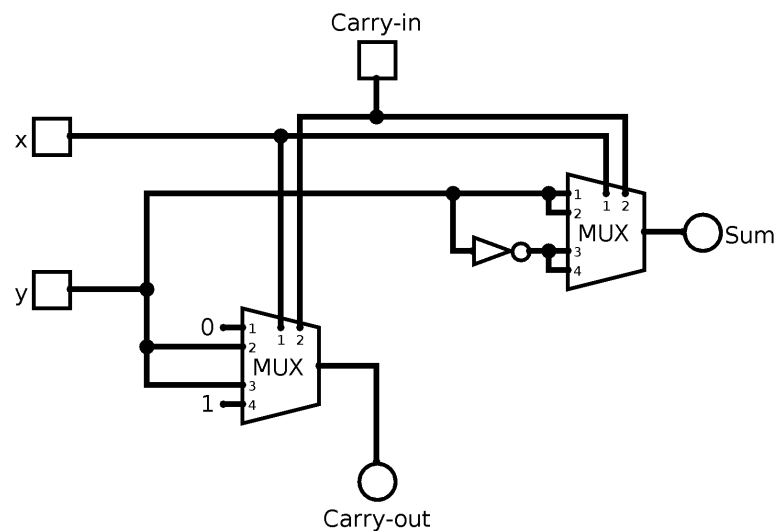


Figure 3: Boole's circuit, containing a mistake

14. André-Marie Ampère and George Boole are participating in the annual Circuit Olympics (CO). One of the challenges is to build a full adder circuit component with only multiplexers and NOT gates. Boole comes up with a design in the blink of a CPU cycle and proudly shows it to Ampère. However, Ampère replies with laughter, as he immediately spots a mistake in Boole's design. What should change in Boole's design (shown in Figure 3) to be a correct full adder?
- A. Switch the select inputs for the multiplexer that outputs the sum
 - B. Switch the select inputs for the multiplexer that outputs the carry-out
 - C. Switch data inputs 1 and 3 for the multiplexer that outputs the sum
 - D. Switch data inputs 2 and 4 for the multiplexer that outputs the sum**

Solution: The solution can be verified by (1) knowing the truth table for both output signals and (2) checking if the circuit implementation is valid using the technique of Figure A.38 from the book.

15. The CO-42 standard, invented last year by CSE1400 instructors to replace IEEE-754, returns this year due to popular request with a few changes. CO-42B works the same way as IEEE-754, with the following specifications:

- There are 20 bits in total,
- the exponent is 8 bits representing the exponent in excess-127,
- the mantissa is 11 bits,
- one sign bit.

What is the decimal value represented by 1 10000110 10010101000.

- A. 101.25
- B. -202.5**
- C. -101.25
- D. -149

Solution: The sign bit is 1, meaning the number is a negative number.

The exponent is 134_{10} , but is in excess-127, so the exponent is really $134 - 127 = 7$.



Shifting by 7 gives $1.10010101_2 \cdot 2^7 = 11001010.1_2$. Converting to decimal gives $0.5 + 2 + 8 + 64 + 128 = 202.5$.

Applying the sign gives -202.5 .

- A shift (by 6) too little makes $1100101.01_2 = 0.25 + 1 + 4 + 32 + 64 = 101.25$.
- Forgetting about the implicit "1." before the mantissa results in $10010101.0_2 = 1 + 4 + 16 + 128 = 149$

16. A student just wrote the following snippet of 64-bit assembly code (AT&T syntax, so the order of the operands is "*source, destination*"):

```

1      foobar:
2          push    %rbp
3          mov     %rsp, %rbp
4
5          push    $8
6          mov     , %rcx
7          mov     , %rax
8          add     %rcx, %rax
9
10         mov     %rbp, %rsp
11         pop     %rbp
12
13         ret
14
15     main:
16         push    $1
17         push    $2
18         push    $4
19         call    foobar

```

Unfortunately, there is a fly sitting on her screen, hiding some memory address at lines 6 and 7. What memory addresses is the fly covering, such that the value 5 ends up in the `%rax` register after the call to `foobar` in the `main` routine?

- A. $-8(\text{\%rbp})$ and $-24(\text{\%rbp})$
- B. $8(\text{\%rbp})$ and $24(\text{\%rbp})$
- C. $-16(\text{\%rbp})$ and $-32(\text{\%rbp})$
- D. $16(\text{\%rbp})$ and $32(\text{\%rbp})$**

Solution: At lines 6 and 7, the stack looks as follows (the top of the stack is on the right, so the the right is negative, to the left positive):

1 2 4 *ret.* *old-rbp* \%rbp 8\%rsp

The value 4 is two places below the base pointer and the value 1 is four places below the base pointer. This is equal to 16 and 32 bytes, respectively.