COMS10015 formative exam #1

This is the question sheet for a formative exam, noting that the choice of each term is important for 2 reasons. First, the term "formative" implies the goal is to improve your understanding of and so performance in the unit: although *not* credit bearing, the feedback (i.e., a mark and a set of example solutions) offered means it should represent an opportunity 1) for additional practice and revision, and 2) to gauge your progress and, e.g., areas of strength and/or weakness. However, second, the term "exam" (versus say "test") implies some level of gravitas: we view it as an important element of the teaching strategy used in the unit, and so engagement with it as mandatory. Note that:

- The exam is taken online (in the sense of being a "take home" exam, for which need to complete and submit a digital answer sheet online), during in one of the consolidation or assessment preparation weeks[a], i.e., *during* TB1 or TB2.
- The exam is open book[b], meaning that use of external resources (e.g., a set of notes) is permitted.
- The exam questions will have a multiple choice[c] format.
- The exam has no fixed duration, beyond the deadline for submitting your answer sheet.

At a high level, there are 3 main tasks (or steps) to consider:

1. **Prepare**.
   - Download the question sheet (i.e., this PDF).
   - Download the (plain text) answer sheet from

     `https://assets.phoo.org/COMS10015_2026_TB-4/csdsp/exam-1/solution.txt`

   - Allocate some time in your schedule to complete the exam.
2. **Complete**. Using the answer sheet, select the options (e.g., those labelled `A`, `B`, `C`, `D`, `E`, and so on) which reflect your answer to each question: do *not* convert the plain text answer sheet into another format (e.g., a PDF, or Word document).
3. **Submit**. Before the stated deadline, submit your completed answer sheet via

   `https://www.ole.bris.ac.uk`

Then, after the deadline, the submitted answer sheets will be (automatically) marked and those marks subsequently released back to you.

[a] See `https://www.bristol.ac.uk/university/dates`.
[b] `https://en.wikipedia.org/wiki/Exam`
[c] `https://en.wikipedia.org/wiki/Multiple_choice`

1. Part #1: Boolean algebra

▷ **Q1.** Consider the Boolean function

$$f : \{0,1\}^n \rightarrow \{0,1\}^m.$$

How many possible input combinations are there for f (resp. how many rows does the truth table for f have)?

`A:` n
`B:` m
`C:` 2
`D:` 2^n
`E:` 2^m

▷ **Q2.** In the list below, which is **not** a standard Boolean axiom?

`A:` de Morgan, i.e., $\neg(x \wedge y) \equiv \neg x \vee \neg y$
`B:` inverse, i.e., $x \wedge \neg x \equiv 0$
`C:` identity, i.e., $x \wedge 1 \equiv x$
`D:` idempotency, i.e., $x \wedge x \equiv x$
`E:` occupancy, i.e., $\neg x \wedge \neg x \equiv x$

▷ **Q3.** Within the following truth table

x	y	A	B	C	D	E
0	0	0	0	0	1	1
0	1	0	1	1	0	0
1	0	0	1	1	0	0
1	1	1	1	0	0	1

which of the labelled, right-hand columns (i.e., A through E) correctly describes $r = x \oplus y$?

▷ **Q4.** From the list below, which Boolean operator would you say is the *best*?

A: AND
B: ANDOR
C: NAND
D: NANDOS

▷ **Q5.** The Boolean expression

$$\neg((w \vee \neg x \vee y \vee \neg z) \vee (x \vee \neg y))$$

is equivalent to which of the following alternatives?

A: 1
B: $(w \vee x)$
C: $\neg(w \wedge x)$
D: $(w \wedge x)$
E: 0

▷ **Q6.** Imagine that you simplify the Boolean expression

$$(\neg(a \vee b) \wedge \neg(c \vee d \vee e)) \vee \neg(a \vee b)$$

with the goal of minimising the total number of operators: if you are allowed to use NOT, AND, and OR only, what is the minimum number required?

A: 1
B: 2
C: 3
D: 4
E: 5

▷ **Q7.** Consider the following 3-input, 1-output Boolean function

$$r = f(x, y, z) = (y \wedge z) \vee (x \wedge z) \vee (x \wedge y).$$

Which of the following best describes f?

A: It produces an r equal to the sum of x, y, and z
B: It produces an r equal to 1 iff. 2 or more of x, y, and z are equal to 1
C: It acts as a multiplexer, selecting between x and y based on z
D: It produces an r equal to the XOR of x, y, and z
E: It produces an r equal to 1 iff. an odd number of x, y, and z are equal to 1

▷ **Q8.** Consider the NAND operator, denoted $\overline{\wedge}$, which you could view as a 2-input, 1-output Boolean function. From the list below, select **each** property which applies to (or is exhibited by) $\overline{\wedge}$.

A: binary
B: constant
C: complete
D: symmetric
E: balanced

`F:` monotone

▷ **Q9.** Consider the following statement. *When converted to Product of Sums (PoS) form, the Boolean expression* $(x \vee y) \wedge (x \oplus z)$ *could read* $(x \vee y) \wedge (x \vee z) \wedge (\neg x \vee \neg z)$. Is the statement

`A:` True
`B:` False

2. Part #2: integer representation and arithmetic

▷ **Q10.** What value does the 8-bit, binary literal 00100111 represent, assuming an unsigned integer representation is used?

`A:` zero
`B:` seven
`C:` eight
`D:` thirty nine
`E:` one hundred thousand, one hundred and eleven

▷ **Q11.** In a C program, you use `0x82` as part of an expression. What decimal value does this literal represent?

`A:` 0
`B:` 16
`C:` 82
`D:` 130
`E:` −126

▷ **Q12.** In a C program, you use `x = 0x82` to assign `x` a value. If the type of `x` is `char`, what decimal value does it have after the assignment?

`A:` 0
`B:` 16
`C:` 82
`D:` 130
`E:` −126

▷ **Q13.** Consider a signed integer, represented in n bits. Let X be the set of values that can be represented if two's-complement is used, and Y be the set of values that can be represented if sign-magnitude is used. Which of the following relations is correct?

`A:` $X \subset Y$ (i.e., X is a sub-set of Y)
`B:` $X = Y$ (i.e., X is equal to Y)
`C:` $X \supset Y$ (i.e., X is a super-set of Y)
`D:` All of the above
`E:` None of the above

▷ **Q14.** Consider the following statement. *If a bit is a base-*2 *digit, a trit would be a base-*3 *digit. Consider an* 8*-bit unsigned integer* x *represented in base-*2*, and an* n*-trit unsigned integer* y *represented in base-*3*:* $n = 6$ *is the smallest choice which will allow* y *to represent all the values* x *can.* Is the statement

`A:` True
`B:` False

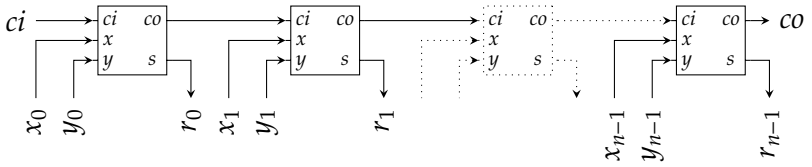
▷ **Q15.** Imagine x and y are unsigned, n-bit integers, whose sum $r = x + y$ can be computed using the following, incomplete algorithm:

```
1 algorithm Add(x, y, n) begin
2 |   r ← 0, c_0 ← 0
3 |   for i = 0 upto n − 1 step +1 do
4 |   |   r_i ← (x_i + y_i + c_i) mod 2
5 |   |   [          ]
6 |   end
7 |   return r
8 end
```

Notice that Line #5 is currently missing: which of the following options would complete the algorithm, and thus allow it to compute the correct result?

A: $c_{i+1} \leftarrow c_i$
B: $c_{i+1} \leftarrow x_i + y_i + c_i$
C: **if** $(x_i + y_i + c_i) < 2$ **then** $c_{i+1} \leftarrow 1$ **else** $c_{i+1} \leftarrow 0$
D: **if** $(x_i + y_i + c_i) < 2$ **then** $c_{i+1} \leftarrow 0$ **else** $c_{i+1} \leftarrow 1$
E: **if** $(x_i + y_i + c_i) < 2$ **then** $c_{i-1} \leftarrow 1$ **else** $c_{i-1} \leftarrow 0$
F: **if** $(x_i + y_i + c_i) < 2$ **then** $c_{i-1} \leftarrow 0$ **else** $c_{i-1} \leftarrow 1$

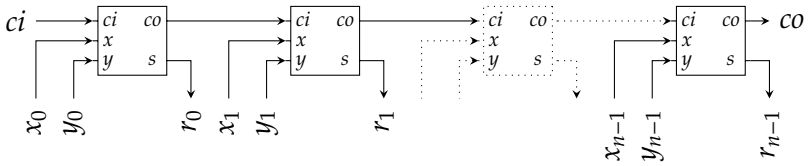
▷ **Q16.** Imagine x and y are signed, n-bit integers for $n = 8$ using two's-complement representation, whose sum $r = x + y$ can be computed using a high level design captured by the following block diagram:



A design engineer, Andy Gate, produces a physical implementation of this design and uses it to compute the sum of $x = 84_{(10)}$ and $y = 94_{(10)}$. When he checks the result, however, he is disappointed to observe $r = -78_{(10)}$ as the result rather than $178_{(10)}$ as expected. Why is this?

A: There is a mistake in the implementation
B: There was a fault during manufacture of the implementation
C: The observed result does not take into account a carry-out
D: The expected result overflowed to produce the observed result

▷ **Q17.** Imagine x and y are unsigned, n-bit integers, whose sum $r = x + y$ can be computed using a high level design captured by the following block diagram:



Each low-level block is a 3-input, 2-output Boolean function, as described by the columns labelled + in the following truth table:

			+		⊙	
ci	x	y	co	s	co	s
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	0	1	0	1
0	1	1	1	0	0	0
1	0	0	0	1	1	1
1	0	1	1	0	1	0
1	1	0	1	0	0	0
1	1	1	1	1	1	1

Imagine the low-level block are swapped for variants described by the columns labelled ⊙: what does the design now compute?

A: It multiplies x by y
B: It adds y to x
C: It subtracts y from x
D: It computes HW(x), i.e., Hamming weight of x
E: It computes HD(x, y), i.e., Hamming distance between x and y

▷ **Q18.** Imagine x and y are unsigned, n-bit integers, which are processed using an algorithm specified as follows:

```
1  algorithm UNKNOWN(x, y, n) begin
2      r ← 0
3      for i = n − 1 downto 0 step −1 do
4          r ← r ≪ 1
5          if y_i = 1 then
6              r ← r + x
7          end
8      end
9      return r
10 end
```

Which of the following descriptions reflects the computation performed by the algorithm?

A: It multiplies x by y
B: It adds y to x
C: It subtracts y from x
D: It computes HW(x), i.e., Hamming weight of x
E: It computes HD(x, y), i.e., Hamming distance between x and y

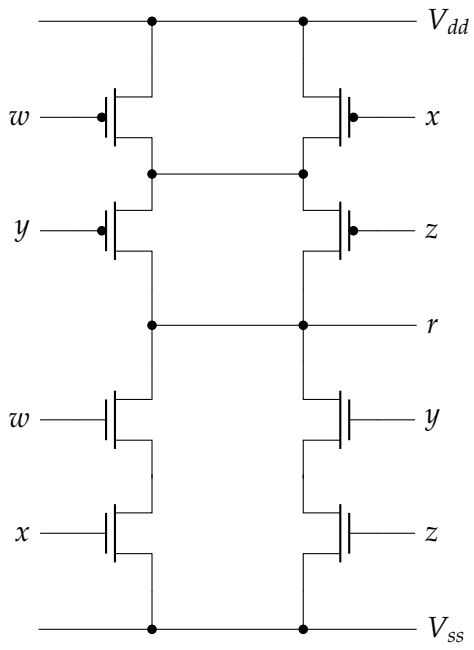
3. Part #3: transistors and logic gates

▷ **Q19.** Within a CMOS-based circuit, which of the following options

	P-MOSFET	N-MOSFET
A	not used	pull-up network
B	pull-down network	not used
C	pull-down network	pull-up network
D	not used	pull-down network
E	pull-up network	not used
F	pull-up network	pull-down network
G	not used	not used

correctly describes the role of constituent transistor types?

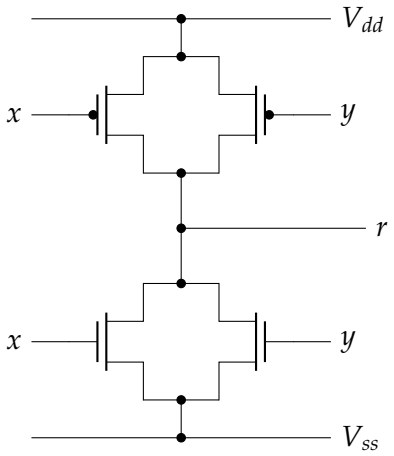
▷ **Q20.** Consider the following CMOS-based circuit:



From the list below, which option describes the functionality implemented by the circuit (resp. the output it produces)?

A: $r = \neg(w \vee x \vee y \vee z)$
B: $r = \neg((w \vee x) \wedge (y \vee z))$
C: $r = \neg(w \wedge x \wedge y \wedge z)$
D: $r = \neg((w \wedge x) \vee (y \wedge z))$

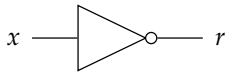
▷ **Q21.** In order to drive the output correctly, the pull-up and pull-down networks within a CMOS-based cell must be complementary; for an output r, satisfying this property will guarantee that $r \in \{0, 1\} \equiv \{V_{ss}, V_{dd}\}$. Consider the following design



which might be described as "partially CMOS", in the sense that the pull-up and pull-down networks are complementary for some input combinations but not others. Assuming $x, y \in \{0, 1\} \equiv \{V_{ss}, V_{dd}\}$, how many input combinations for this design satisfy the complementary property demanded by CMOS-based cells?

A: 0
B: 1
C: 2
D: 3
E: 4

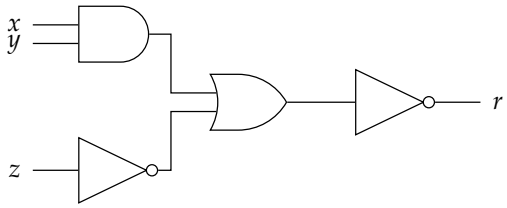
▷ **Q22.** When we express a design in terms of logic gates rather than transistors, e.g., as in



why are the power rails no longer shown?

A: They are no longer required
B: They are no longer relevant
C: Doing so is too much effort

▷ **Q23.** Imagine the gate delay for NOT, AND, and OR gates is 10ns, 20ns, and 20ns respectively. Based on this information, compute the critical path associated with the following circuit:

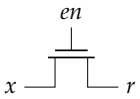


A: 10ns
B: 20ns
C: 30ns
D: 40ns
E: 50ns
F: 60ns

▷ **Q24.** Within the following truth table

x	en	A	B	C	D	E
0	0	0	1	**Z**	**Z**	0
0	1	0	1	1	0	1
1	0	1	0	**Z**	**Z**	1
1	1	1	0	0	1	0

which of the labelled, right-hand columns (i.e., A through E) correctly describes the output r shown in the following circuit:



4. Part #4: combinatorial logic

▷ **Q25.** The sentence

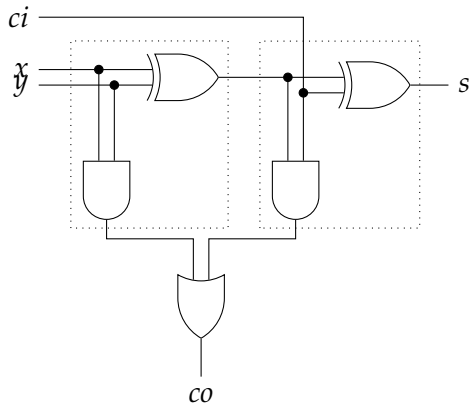
"the output is a function of ① *, and computation is viewed as being performed* ② *"*

might be used to describe a given combinatorial logic design iff. the missing, labelled parts were completed. Within the following table

	①	②
A	a combination of events	recursively, i.e., at a time determined by the previous computation
B	the input only	continuously, i.e., all the time
C	the input plus any state	discretely, i.e., one step at a time
D	the value of I/O pins	randomly, i.e., at some time
E	the state only	logically, i.e., at whatever time makes sense

which of the labelled rows (i.e., A through E) provides the correct completions?

▷ **Q26.** The following design



is a full-adder. Your friend, Ian Teger, claims that the design highlights use of isolated replication: his rationale is that the design stems from use of two half-adder instances (annotated via two dashed boxes). Do you think he is correct?

`A:` No, he is incorrect
`B:` Yes, he is correct

▷ **Q27.** Consider a 3-input, 1-output Boolean function f, for which we aim to express as a Boolean expression using a Karnaugh map. From the list below, identify **each** claim that applies or should be followed while doing so.

`A:` The Karnaugh map itself will be either a (2×4)- or (4×2)-element grid
`B:` Each element in the grid containing 1 must be covered by a group
`C:` No element in the grid containing 0 should be covered by a group
`D:` A given group must have a rectangular shape, and cover $m = 2^{m'}$ elements for some $m' \geq 0$
`E:` Using fewer groups will yield a more optimised result
`F:` Using larger groups will yield a more optimised result

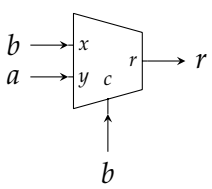
▷ **Q28.** Consider the following truth table

x	y	z	r
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

which describes a 3-input, 1-output Boolean function $r = f(x, y, z)$. If we translate this truth table into a Karnaugh map, and then into a Boolean expression, which of the following describes the groups used (assuming we use the fewest number of, and largest groups possible)?

`A:` 1 group covering 1 element and 1 group covering 4 elements
`B:` 3 groups each covering 2 elements
`C:` 1 group covering 1 element and 2 groups each covering 2 elements
`D:` 1 group covering 5 elements
`E:` 5 group each covering 1 element

▷ **Q29.** Consider the following, single-component combinatorial logic design:



Which of the following Boolean expressions correctly describes the functionality realised?

A: $r = 0$
B: $r = a \vee b$
C: $r = a \wedge b$
D: $r = a \oplus b$
E: $r = 1$

▷ **Q30.** Consider the following statement. *A demultiplexer can be used as a form of decoder.* Is the statement

A: True
B: False