Q1. a Identify **each** statement which correctly describes an N-type MOSFET:

i Has N-type semiconductor terminals and P-type body.

ii Has P-type semiconductor terminals and N-type body.

iii Has a threshold voltage above which the transistor is deemed active.

iv Is paired with another N-type MOSFET to form a CMOS cell.

b Identify **each** component which would **not** typically be used in a combinatorial circuit design:

i Ripple-carry adder.

ii D-type flip-flop.

iii Buffer.

iv Clock.

c Write the simplest (i.e., with **fewest** operators) possible Boolean expression that implements the Boolean function

$$r = f(x, y, z)$$

described by

f			
x	y	z	r
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	**?**
1	0	0	1
1	0	1	0
1	1	0	**?**
1	1	1	1

where **?** denotes don't care.

d Take the Boolean expression

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

and draw a gate-level **circuit diagram** that computes an equivalent resulting using only 2-input NAND gates.

e Recall that an SR latch has two inputs S (or set) and R (or reset); if $S = R = 1$, the two outputs Q and $\neg Q$ are undefined. This issue can be resolved by using a reset-dominate latch: the alternative design has the same inputs and outputs, but resets the latch (i.e., has $Q = 0$ and $\neg Q = 1$) whenever $S = R = 1$.

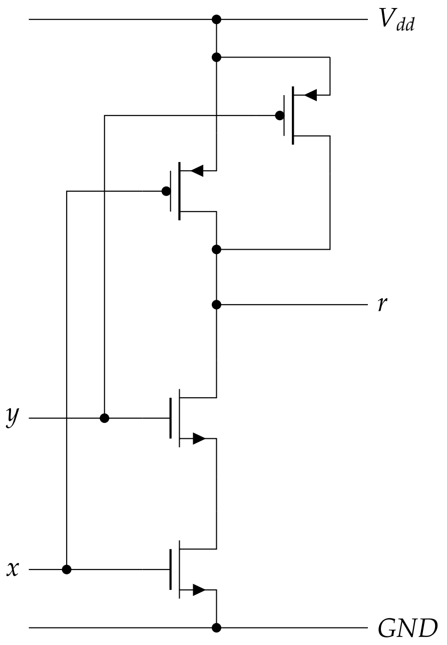
Using a gate-level **circuit diagram**, describe how a reset-dominate latch can be implemented using **only** NOR gates and at most **one** AND gate.

f The quality of the design for some hardware component is often judged by measuring efficiency, for example how quickly it can produce output on average. Name **two** other metrics that might be considered.

Q2. a Describe how N-type and P-type MOSFET transistors are constructed using silicon and how they operate as switches.

b Draw a diagram to show how N-type and P-type MOSFET transistors can be used to implement a NAND gate. Show your design works by describing the transistor states for each input combination.

Q3. The following diagram



details a 2-input NAND gate comprised of two P-MOSFET transistors (top) and two N-MOSFET transistors (bottom). Draw a similar diagram for a 3-input NAND gate.

Q4. Moore's Law predicts the number of CMOS-based transistors we can manufacture within a fixed sized area will double roughly every two years; this is often *interpreted* as doubling computational efficiency over the same period. Briefly explain **two** limits which mean this trend cannot be sustained indefinitely.

Q5. Given that **?** is the don't care state, consider the following truth table which describes a function p with four inputs (a, b, c and d) and two outputs (e and f):

p					
a	b	c	d	e	f
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	**?**	**?**
0	1	0	0	0	1
0	1	0	1	1	0
0	1	1	0	0	0
0	1	1	1	**?**	**?**
1	0	0	0	1	0
1	0	0	1	0	0
1	0	1	0	0	1
1	0	1	1	**?**	**?**
1	1	0	0	**?**	**?**
1	1	0	1	**?**	**?**
1	1	1	0	**?**	**?**
1	1	1	1	**?**	**?**

a From the truth table above, write down the corresponding Sum of Products (SoP) equations for e and f.

b Simplify the two SoP equations so that they use the minimum number of logic gates possible. You can assume the two equations can share logic.

Q6. Using a Karnaugh map, derive a Boolean expression for the function

$$r = f(x, y, z)$$

described by the truth table

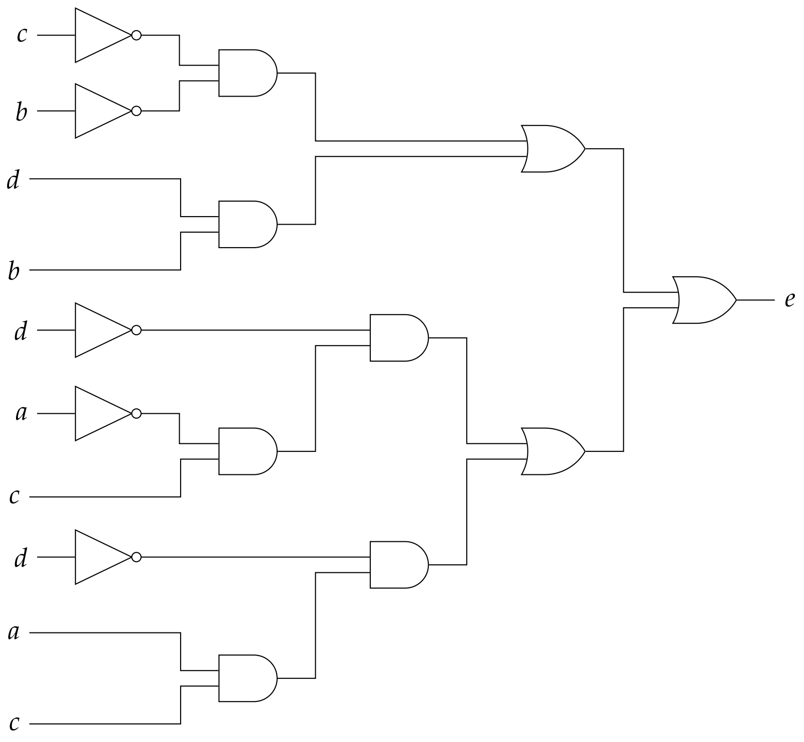
f			
x	y	z	r
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	**?**

where **?** denotes don't care.

Q7. NAND is a universal logic gate in the sense that the behaviour of NOT, AND and OR gates can be implemented using only NAND. Show how this is possible using a truth table to demonstrate your solution.

Q8. Both NAND and NOR gates are described as universal because *any* other Boolean gate (i.e., AND, OR, NOT) can be constructed using them. Imagine your friend suggests a 4-input, 1-bit multiplexer (that selects between four 1-bit inputs using two 1-bit control signals to produce a 1-bit output) is also universal: state whether or not you believe them, and explain why.

Q9. Consider the following circuit where the propagation delay of logic gates in the circuit are 10 ns for NOT, 20 ns for AND, 20 ns for OR and 60 ns for XOR:

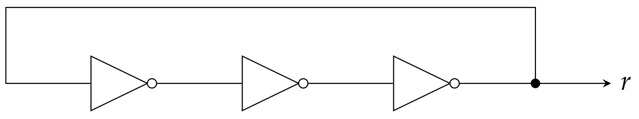


a Draw a Karnaugh map for this circuit and derive a Sum of Products (SoP) expression for the result.

b Describe advantages and disadvantages of your SoP expression and the dynamic behaviour it produces.

c If the circuit is used as combinatorial logic within a clocked system, what is the maximum clock speed of the system?

Q10. Consider the following circuit:

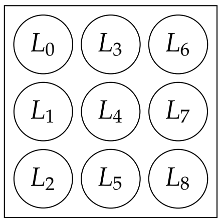


a Describe **two** features that might cause someone to describe this design as strange.

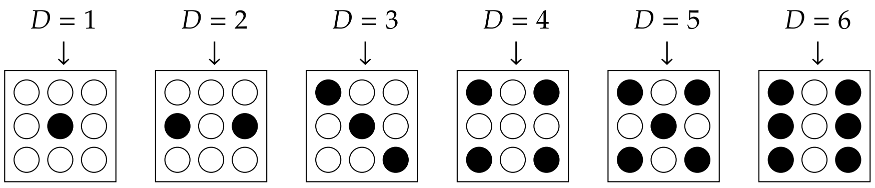
b Describe a purpose for the design, i.e., what the circuit might be used for.

Q11. A game uses nine LEDs to display the result of rolling a six-sided dice; the i-th LED, say L_i for $0 \leq i < 9$, is driven with 1 or 0 to turn it on or off respectively. A 3-bit register D represents the dice as an unsigned integer.

a The LEDs are arranged as follows,



and the required mapping between dice and LEDs, given a filled dot means an LED is on, is



Using Karnaugh maps as appropriate, write a simplified Boolean expression for **each** LED (i.e., for **each** L_i in terms of D).

b The 2-input XOR, AND, OR and NOT gates used to implement your expressions have propagation delays of 40, 20, 20 and 10 nanoseconds respectively. Calculate how many times per-second the dice can be rolled, i.e., D can be updated, if the LEDs are to provide the correct output.

c The results of individual dice throws will be summed using a ripple-carry adder circuit, to give a total; each 3-bit output D will be added to and stored in an n-bit accumulator register A.

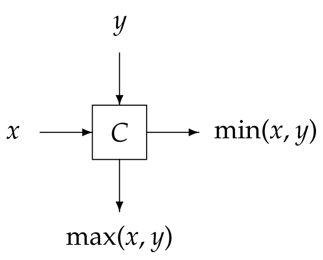
i Using a high-level **block diagram**, show how an n-bit ripple-carry adder circuit is constructed from full-adder cells.

ii If $m = 8$ throws of the dice are to be summed, what value for n should be selected?

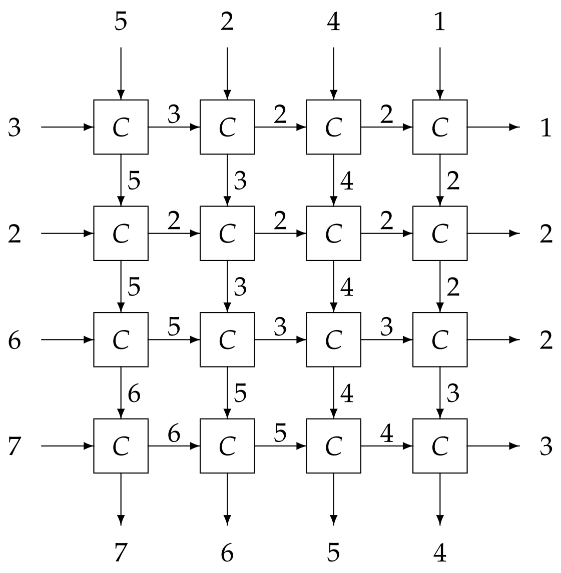
iii Based on the gate propagation delays above, estimate the total propagation delay of an n-bit ripple-carry adder.

iv Imagine that instead of D, we want to add $2 \cdot D$ to A. Doubling D can be achieved by computing either $D + D$ or $D \ll 1$ (i.e., a left-shift of D by 1 bit). Carefully state which method is preferable, and why.

Q12. Consider a simple component called C that compares two inputs x and y (both are unsigned 8-bit integers) in order to produce their maximum and minimum as two outputs:



Instances of C can be connected in a mesh to sort integers: the input is fed into the top and left-hand edges of the mesh, the sorted output appears on the bottom and right-hand edges. An example is given below:



a Using standard building blocks (e.g., adder, multiplexer etc.) rather than individual logic gates, draw a **block diagram** that implements the component C.

b Imagine that an $n \times n$ mesh of components is created. Based on your design for C and clearly stating any **assumptions** you need to make, write down an expression for the critical path of such a mesh.

c Algorithms for sorting integers can clearly be implemented on a general-purpose processor. Explain **two** advantages and **two** disadvantages of using such a processor versus using a mesh like that above.

Q13. Imagine you are working for a company developing the "Pee", a portable games console. The user interface is a fancy controller that has

- three fire buttons represented by the 1-bit inputs F_0, F_1 and F_2, and
- a 8-direction D-pad represented by the 3-bit input D

and you are charged with designing some aspects of it.

a The fire button inputs are described as level triggered and active high; explain what this means (in comparison to the alternatives in each case).

b Some customers want an "autofire" feature that will automatically and repeatedly press the F_0 fire button for them. The autofire can operate in four modes, selected by a switch called M: off (where the fire button F_0 works as normal), slow, fast or very fast (where the fire button F_0 is turned on and off repeatedly at the selected speed). Stating any assumptions and showing your working where appropriate, design a circuit that implements such a feature.

c In an attempt to prevent counterfeiting, each controller can only be used with the console it was sold with. This protocol is used:

$\mathcal{P}$		$\mathcal{C}$
$c \xleftarrow{\$} \{0,1\}^3$		
	$\xrightarrow{c}$	
		$r = T(c)$
	$\xleftarrow{r}$	
$r \overset{?}{=} T(c)$		

which, in words, means that

- the console generates a random 3-bit number c and sends it to the controller,
- the controller computes a 3-bit result $r = T(c)$ and sends it to the console,
- the console checks that r matches $T(c)$ and assumes the controller is valid if so.

i There is some debate as to whether the protocol should be synchronous or asynchronous; explain what your recommendation would be and why.

ii The function T is simply a look-up table. For example

$$T(x) = \begin{cases} 2 & \text{if } x = 0 \quad 4 \quad \text{if } x = 4 \\ 6 & \text{if } x = 1 \quad 0 \quad \text{if } x = 5 \\ 7 & \text{if } x = 2 \quad 5 \quad \text{if } x = 6 \\ 1 & \text{if } x = 3 \quad 3 \quad \text{if } x = 7 \end{cases}$$

Each pair of console and controller has such a T fixed inside them during the manufacturing process. Stating any assumptions and showing your working where appropriate, explain how **this** T might be implemented as a circuit.

Q14. Imagine you have three Boolean values x, y, and z. Given access to **as many** AND and OR gates as you want but **only two** NOT gates, write a set of Boolean expressions to compute all three results $\neg x$, $\neg y$ and $\neg z$.

Q15. SAT is the problem of finding an assignment to n Boolean variables which means a given Boolean expression is satisfied, i.e., evaluates to 1. For example, given $n = 3$ and the expression

$$(x \wedge y) \vee \neg z,$$

$x = 1, y = 1, z = 0$ is one assignment (amongst several) which solves the associated SAT problem.

The ability to solve SAT can be used to test whether or not two n-input, 1-output combinatorial circuits C_1 and C_2 are equivalent. Show how this is possible.

Q16. Consider the following combinatorial circuit, which is the composition of four parts (labelled A, B, C and REG): each part is annotated with a name and an associated critical path. The circuit computes an output $r = f(x)$ from the corresponding input x.

$$\begin{tikzcd} x \arrow[r] & \boxed{\begin{array}{c} A \\ 10\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} B \\ 30\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} C \\ 20\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} REG \\ 10\text{ ns} \end{array}} \arrow[r] & r = f(x) \end{tikzcd}$$

With respect to this circuit,

a first define the terms latency and throughput, then

b explain how and why you would expect use of pipelining to influence both metrics.

Q17. The figure below shows a block of combinatorial logic built from seven parts; the name and latency of each part is displayed inside it. Note that the last part is a register which stores the result:

$$\begin{tikzcd} x \arrow[r] & \boxed{\begin{array}{c} A \\ 40\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} B \\ 10\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} C \\ 30\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} D \\ 10\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} E \\ 50\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} F \\ 10\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} REG \\ 10\text{ ns} \end{array}} \arrow[r] & r = f(x) \end{tikzcd}$$

It is proposed to pipeline the block of logic using two stages such that there is a pipeline register in between parts D and E:

$$\begin{tikzcd} x \arrow[r] & \boxed{\begin{array}{c} A \\ 40\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} B \\ 10\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} C \\ 30\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} D \\ 10\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} REG \\ 10\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} E \\ 50\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} F \\ 10\text{ ns} \end{array}} \arrow[r] & \boxed{\begin{array}{c} REG \\ 10\text{ ns} \end{array}} \arrow[r] & r = f(x) \end{tikzcd}$$

a Explain the terms latency and throughput in relation to the idea of pipelining.

b Calculate the overall latency and throughput of the initial circuit described above.

c Calculate the overall latency and throughput of the circuit after the proposed change.

d Calculate the number of extra pipeline registers required to maximise the circuit throughput; state this new throughput and the associated latency. Explain the advantages and disadvantages of this change.