1 Boolean Functions (10+10+15 = 35 points)

1.1 Unate Functions

A Boolean function $f(x_1, \ldots, x_n)$ is said to be *positive unate* in input variable x_i if changing x_i from 0 to 1, while keeping the other inputs constant at any of their possible values, can never make f go from 1 to 0. In other words,

$$f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n) \geq f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n)$$
$$\forall \; x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$$

Similarly, a Boolean function is *negative unate* in input variable x_i if changing x_i from 0 to 1 can never make f go from 0 to 1. A function that is neither positive unate nor negative unate in an input variable is said to be *binate* in that variable. A function that is unate in all its input variables is called a *unate function.*

Determine whether the following functions are positive unate, negative unate, or binate in **each** of their input variables. Justify your answers.

10 points The majority function: $Majority(x_1, x_2, \ldots, x_n) = 1$ if and only if more inputs are 1 than 0 (assume that n is odd).

10 points The exact-count function: $ExactCount_k(x_1, x_2, \ldots, x_n) = 1$ if and only if exactly k of the inputs are 1. Note that k is a constant, *i.e.*, it is not an input to the function.

1.2 Symmetric Functions

A *symmetric* Boolean function is a Boolean function whose value does not change under any permutation of the input values, *i.e.*, it only depends on the number of inputs that assume values of 0 and 1. The majority and exact-count functions given in Section 1.1 are examples of symmetric functions. For example, the majority function depends only on how many inputs are 1, and not on which specific inputs are 1.

Question (15 points): How many distinct **symmetric** Boolean functions exist with n inputs and 1 output? Justify your answer.

Write in Exam Book Only

2 Logic minimization (10+15=25 points)

Boolean functions may be represented as *sum-of-products expressions*, and implemented using two-level logic. A product term is also called an *implicant*. To minimize hardware requirments, it is desirable to have *prime implicants*. A prime implicant is not contained in any other implicant of the function. *Essential prime implicants* are especially useful since they uniquely cover a minterm that is not covered by any other prime implicant of the function.

For the special case of unate functions (defined in the previous section), the process of identifying essential primes is greatly simplified.

Answer the following questions:

10 points. Show that the following function is unate in all its input variables:
$f(a, b, c, d, e) = ab + ace + bcd + abce + cde + bc'e + a'bce$

15 points. Find all essential primes for this function. For each essential prime, specify a minterm that makes it essential (*i.e.*, a minterm that is covered by the prime and no other prime). Give your answers by copying the following table into your answer booklet and completing it. You will not get credit for answers written on the question paper!

Essential Prime	Essential minterm

3 Timing (20+20=40 points)

A logic circuit's clock period is determined by the longest time or delay that the combinational logic can take to produce its outputs once its inputs are applied. It is common to

estimate the delay of a combinational circuit as the length of its longest path(s). However, in general, this estimate may be pessimistic due to the presence of *false paths*, or paths that can never affect the circuit's delay.

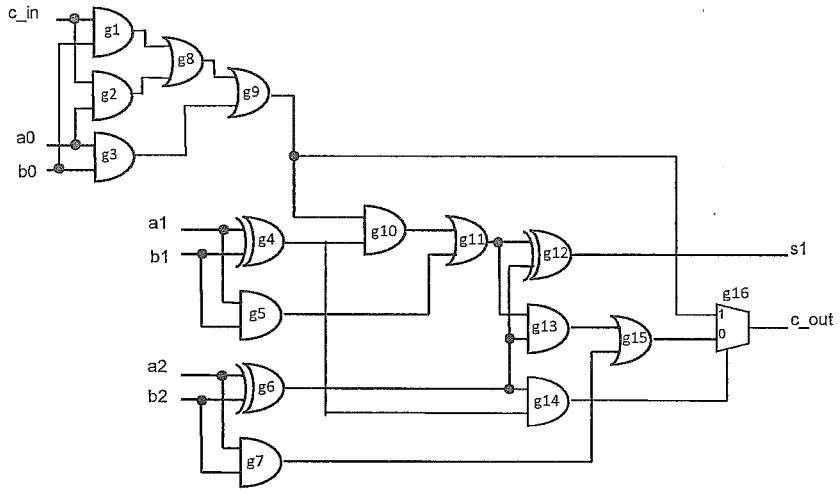

Figure 1: A combinational circuit

Consider the circuit given in Figure 1. Assume that (i) each gate has a delay of 1 unit, and (ii) a path's delay is the sum of the delays of the gates along the path. Answer the following questions:

20 points. What is the length of the longest path in this circuit? Identify all longest paths.

20 points. For each of the longest paths, state whether or not it is a false path. Justify your answer.

Write in Exam Book Only