Only Problem Set Part B will be graded. Turn in only Problem Set Part B which will be due on October 25, 2019 (Friday) at 9:00 am.

1 Problem Set Part A

All questions in this part are from your textbook by Bushnell & Agrawal.

- 1.1, 1.4
- 4.2, 4.4, 4.5, 4.8, 4.9, 4.10, 4.11, 4.12
- 7.1, 7.2, 7.3, 7.4, 7.6, 7.7, 7.8, 7.10, 7.11, 7.15, 7.17, 7.19, 7.26
- 6.3, 6.5, 6.7
2 Problem Set Part B

I. (Functional Fault Equivalence)

The sum-of-products formulation of an XOR function can be written as \(A'B + AB'\). You can see that there exist 22 stuck-at faults in this 2 2-input AND, 1 2-input OR and 2 INVERTER implementation.

(Part A) Please identify equivalent faults within this set of 22 faults which can be obtained solely through functional equivalence.

(Part B) The XNOR function can be written in a product-of-sums form in a rather similar representation as \((A' + B) \times (A + B')\), implemented in 2 2-input OR, 1 2-input AND and 2 INVERTER form. Please identify equivalent faults within this set of 22 faults which can be again obtained solely through functional equivalence.

(Part C) The XOR function can be written in a product-of-sums form as well, albeit in a slightly different formulation as \((A' + B') \times (A + B)\), implemented in 2 2-input OR, 1 2-input AND and 2 INVERTER form. Please identify equivalent faults within this set of 22 faults which can be obtained solely through functional equivalence.
II. (D-algorithm)

The carry and sum functions for a single-bit adder are defined as:

\[
\begin{align*}
\text{sum} &= x \oplus y \oplus c_{in} \\
\text{cout} &= xy + xc_{in} + yc_{in}
\end{align*}
\]

Behaviorally speaking, a carry (\(c_{out}\)) has the value of 1 if 2 or more of the three input bits to a single bit adder (\(x, y, c_{in}\)) are set, while the sum (\(\text{sum}\)) is 1 if an odd number of the three inputs is set.

In the context of this question, these (\(\text{sum}\)) and (\(c_{out}\)) functions are shown in the design as single components, necessitating the development of D-algorithm procedures to handle them. This question asks you to extend your understanding of D-algorithm procedures that we discussed in the context of 2-input Boolean gates to such slightly larger Boolean functions.

(Part A) Please define the singular covers for the (\(\text{sum}\)) and (\(c_{out}\)) functions.

(Part B) Using the singular covers, please derive the propagation D-cubes for the (\(\text{sum}\)) and (\(c_{out}\)) functions.

(Part C) The Fault Modeling team has identified a new fault type that has been plaguing the carry (\(c_{out}\)) function implementation. Apparently, the carry (\(c_{out}\)), in an inexplicable way, is manifesting a tendency of mutating to the (\(\text{sum}\)) function. Please develop primitive D-cubes of failure for this fault by using cube sets \(\alpha_0, \alpha_1, \beta_0,\) and \(\beta_1\).
III. (Advanced Combinational Test)

In this question you will be working on this circuit that should be familiar to you from our discussions in class.

(Part A) Please identify all *static learning* implications that SOCRATES will identify. Please refrain from including implications that will be straightforwardly derived through local propagation rule application.

(Part B) Please identify any *headlines* in this circuit. Please show how FAN will utilize such headlines in resolving a test for the s-a-0 fault at the primary output (i.e. line n).