EECS 579 Homework No.1 Solutions

Problem 1 (15 points) *Defect level* Text, Page 16, Problem 1.1. Also give your answer in ppm.

In general, a "yield" parameter y_T is specified with respect to some test (set) T, and is the probability that a chip passes T or, equivalently, y_T is the percentage of chips that T says are fault-free. The accuracy of y_T depends on the quality of T. A bad test T will produce yield numbers that are too high (in which case defective parts are delivered to customers) or too low (in which case good parts are thrown away). In either case, bad tests are costly to the chip manufacturer. The "true" yield is what perfect testing would produce.

For this problem, the events of Example 1.1 are redefined as follows: PQ = chip is good; P = chip passes the test; FQ = chip is bad; and F = chip fails the test. A 70% true yield means Prob(PQ) = 0.7 and Prob(FQ) = 0.3. Following the analysis of Example 1.1 (Eq. 1.1 with the same numbers), we get Prob(P) = 0.68. Hence,

Defect level = Bad chips that pass the tests / All chips that pass the tests
= [Prob(P|FQ) * Prob(FQ)] / Prob(P)
= 0.05 * 0.3 / 0.68 = 0.022, which is 22,000 ppm.

Problem 2 (15 points) *Defect coverage* Text, Page 16, Problem 1.2.

As in Prob.1, a defect level of 500 ppm means [Prob(P|FQ) * Prob(FQ)] /Prob(P) = 0.0005. Now the denominator Prob(P) = Prob(P|PQ) * Prob(PQ) + Prob(P|FQ) * Prob(FQ). So if we let x denote the escape probability Prob(P|FQ), we can write the defect level as

[Prob(P|FQ)Prob(FQ)] /[Prob(P|PQ) * Prob(PQ) +Prob(P|FQ) * Prob(FQ)]

$= x * 0.3/ [0.95 * 0.7 + x * 0.3] = 0.0005$

which implies $x = 0.0003325//0.29985$. Next, we have

Defect coverage = Prob(F|FQ) = 1 – Prob(P|FQ) = $1 - x = 0.99889$

The required defect coverage is thus 99.889%.

Problem 3 (15 points) *Testing cost* Text, Page 16, Problem 1.4

Following Example 1.2, we obtain

ATE purchase price = \$1.2M + 256 * \$3,000 = \$1.968M

Assuming a 20% per year linear rate of depreciation, a maintenance cost of 2% of the purchase price, and an annual operating cost of \$0.5M, we get

Running cost = \$1.968M * 0.2 + \$1:968M * 0.02 + \$0.5M = \$932,960/year

Testing cost = \$932,960/(365 x 24 * 3600) = 2.96 cents/second

The testing cost of the self-test design is thus 2.96 cents per second, which is down from the 4.50 cents per second calculated in Example 1.2

Problem 4 (15 points) *Benefit-cost analysis* Text, Page 55, Problem 3.3 (as corrected)

(a) *Complete elimination of burn-in*: Let C_t be the total cost of a chip where a burn-in test is applied to every chip that passes conventional testing. Let C_b be the per-chip cost of burn-in testing. C_t includes C_b, as well as another component C_f, which accounts for the costs of fabrication, conventional test, etc. The total cost per chip is given by,

$C_t = (C_f + y_c C_b)/(y_c y_b)$

where y_c is the yield of the conventional test and y_b is the yield of the burn-in test. Since the cost of the

I_{DDQ} test is 10% of the burn-in cost and there is a 10% yield loss, the total cost per chip when burn-in testing is replaced by I_{DDQ} testing is given by,

$$C'_t = (C_f + 0.1y_cC_b)/(0.9y_cy_b)$$

For this new scheme to be beneficial, we must have $C'_t < C_t$. This implies

$$(C_f + 0.1y_cC_b)/(0.9y_cy_b) < C_t$$

We now manipulate the left-hand side of this inequality to get it in terms of C_t.

$$(C_f + 0.1y_cC_b)/(y_cy_b) < 0.9C_t$$

$$[(C_f + y_cC_b - 0.9y_cC_b)]/(y_cy_b) < 0.9C_t$$

$$(C_f + y_cC_b)/(y_cy_b) - 0.9y_cC_b/(y_cy_b) < 0.9C_t$$

$$C_t - 0.9y_cC_b/(y_cy_b) < 0.9C_t$$

Hence we can write

$$0.1C_t < 0.9y_cC_b/(y_cy_b)$$

or, equivalently,

$$C_t < 9C_b/y_b$$

We are told that the burn-in yield $y_b = 0.9$, therefore $C_t < 10C_b$. It follows immediately that the total cost should not exceed ten times the burn-in cost, as stated in the problem.

(b) *Apply burn-in test only to chips that fail IDDQ test*: Suppose we apply the I_{DDQ} test to all chips passing the pre-burn-in test. Due to the 10% yield loss relative to the burn-in yield, this will produce a yield of $0.9y_b$. The remaining fraction, $1 - 0.9y_b$ must be subjected to the burn-in test to recover the lost yield. For the new scheme to be beneficial, we must have

$$0.1C_b + (1 - 0.9y_b)C_b < C_b$$

or $y_b > 1/9$. Hence the burn-in yield should be greater than 1/9 or 11.1%.

Problem 5 (10 points) *Fault counting* Text, Page 79, Problem 4.4.

For the circuit of Figure 4.6 (p. 72), we have

No. of fault sites = No. of PIs + No. of gates + No. of fanout branches = 2 + 4 + 6 = 12

Therefore, the number of multiple (MSL) faults = $3^{\text{No. of fault sites}} - 1 = 3^{12} - 1 = 531{,}440$. Note that this figure includes all the single faults.

Problem 6 (15 points) *General fault analysis*

(a) Two faults are equivalent if their columns in the fault table are identical. Clearly the only such pairs are $f_1 = f_5$ and $f_2 = f_4$. Hence there are six equ ivalence classes: $\{f_0\}$, $\{f_1,f_5\}$, $\{f_2,f_4\}$, $\{f_3\}$, $\{f_6\}$, $\{f_7\}$. Some people omitted the four singleton (one-member) classes.

(b) There are 15 pairs f_if_j of nonequivalent fault classes to distinguish. We can set up a 6-by-15 fault table and find a minimum cover in the usual way. No single test vector covers this table (distinguishes all 15 pairs). The test sets $\{t_1,t_6\}$ and $\{t_2,t_4\}$ are covers and so are the minimal distinguishing test sets.

(c) Now we are told that f_0 represents the correct or fault-free response. Hence, we just need to cover the 6-by-5 subtable consisting of all pairs of the form f_0f_j. From this we can quickly see that any one of the tests t_2, t_5 or t_6 will detect all faults.

Problem 7 (15 points) *Gate faults and tests* 5-input NOR gate

(a) NOR5 has 6 fault sites and therefore $3^6 - 1 = 728$ distinct MSL faults, all of which are detectable.

(b) The number of MSL classes of equivalent faults in NOR5 is $2^5 = 32$. To show this, it is definitely *not* necessary to compare all the 728 possible MSL faults. It is much more efficient to identify, and examine, a small but complete set of representative cases.

Case 1: *Multiple faults where the output line z is stuck-at-d and any combination of input lines are faulty*: For a fixed *d*, all such multiple faults must be equivalent since they effectively change the gate's output function to $z(x_1,x_2,x_3,x_4,x_5) = d$. Hence there are just two classes with representative faults $z/0$ and $z/01$.

In all the remaining cases, we can assume that the output line is fault-free.

Case 2: *Faults where the output line is fault-free, but k input lines, $1 \leq k \leq 5$, are stuck-at-1*: All such faults make $z = 0$ and so belong to one of the two classes of Case 1. So far, we have identified just two (big) sets of equivalent faults.

Case 3: *Faults where the output line is fault-free, no input lines are stuck-at-1, but some k input lines are stuck-at-0, for* $0 \leq k \leq 5$. Now there are $2^5 = 32$ subsets of *k* lines so we have 32 possible faults. When $k = 5$, we have $z = 0$, which is covered by Case 1. When $k = 0$, we have the fault-free case. It's easy to see that the remaining 30 cases are all distinguishable from each other. (Check that this is so.) Since we have covered all possibilities, combining these 30 nonequivalent classes with the two identified in Case 1, yields a grand total of 32 MSL classes.

(c) It's easily shown that the unique minimal set of 6 tests that detect all SSL faults in NOR5 also detect all MSL faults in NOR5. These tests are x_1,x_2,x_3,x_4,x_5 = 00000,10000,01000,00100,00010,00001.

Exercise: Generalize these results of Problem 7 to *k*-input (elementary) gates for any $k \geq 1$.