**Q1:** A full AES-128 encryption completes when a ciphertext is ready. Then how many encryptions did this testbench test?

**A1:** 4 encryptions. Because the ‘ld’ is raised 4 times, so did the ‘done’ signal.

**Q2:** How many clock cycles did one encryption take?

**A2:** 12 cycles.

**Q3:** If the ‘done’ signal is forced to be raised one cycle earlier than its current raising cycle. Do you think this AES encryption is still secure? Why?

**A3:** It is still secure, because the AES has gone through 9 rounds of encryption, even though the pre-mature ciphertext becomes available, it is still difficult to break the AES based on the 9th round intermediate states.

**Q4:** Under what circumstances do you think the AES is not secure? Why? (This is an open question)

**A4:** When the ‘done’ signal is raised too early due to fault-injections, e.g., within the 1st round. Attackers could derive the key by simply XORing the intermediate state and plaintext.

Other possible answers: when the 8th or 9th round intermediate states or round key bytes are faulty. These are the typical targets of Differential Fault Analysis (DFA) attacks. In general, faults on these locations could facilitate attackers to calculate the secret key based on a limited number of good and faulty ciphertext pairs.

**Q5:** Is SP1 rigorous enough to define the attack scenario in Section 1.3? Why? If not, could you write another security property file to define this scenario?

**A5:** No. It only checks one cycle during the early cycles after ‘ld’ is raised. See strobe\_2.sv in lab\_db.zip for an example.

**Q6:** How many faults this .sff line in contains?

NA ~ (25^26) { "done\_fanin.done\_reg.Q" + "done\_fanin.dcnt\_reg\_0\_.Q" }

**A6:** 1 fault. It has two targets flipping simultaneously, it’s a double-fault.

**Q7:** How many faults this .sff line contains?

NA ~ (22^23, 23^24, 24^25, 26^27, 27^28) { "done\_fanin.done\_reg.Q" + "done\_fanin.dcnt\_reg\_2\_.Q" + "done\_fanin.dcnt\_reg\_1\_.Q" }

**A7:** 5 faults.

**Q8:** Do we need to cover a longer fault-injection window to include cycles > 27? Why?

**A8:** No. Because the signal checking of the security property (the “perfect fault”) happens at cycle 27. Any faults injected later could never violate the property.

**Q9:** Are faults on combinational cells included in this fault list? If not, can you develop one?

**A9:** No. Here is an example covering comb. cells:

NA ~ (1^2, 2^3, 3^4, 4^5, 5^6, 6^7, 7^8, 8^9, 9^10, 10^11, 11^12, 12^13, 13^14, 14^15, 15^16, 16^17, 17^18, 18^19, 19^20, 20^21, 21^22, 22^23, 23^24, 24^25, 25^26, 26^27, 27^28) { PORT "done\_fanin.\*.Y" }

**Q10:** Based on the background knowledge in Section 1.1, can you briefly describe under what fault-injection scenario we should include combinational cells into fault lists?

**A10:** We should add both seq. and comb. cells to fault lists when assuming an EM fault-injection scenario or an optical (e.g., laser) fault-injection scenario, because these attack categories possess the potential to precisely flip any cell’s value in the circuit. While clock glitching and voltage glitching attacks create faults primarily due to inducing timing violations.