

Solution of Assignment 2.

8.3.

a) n in C , a in D
 m in C

b) Please give more description of the execution.

Ex: step 1
since $\left\{ \begin{array}{l} \# \text{define } \text{FOO } 0x0010 \\ m = (\text{int}^*) \text{FOO}; \\ \text{the content at } 0x0010 \text{ is } 0 \end{array} \right.$

So $*m = 0$

step 2

$\text{foo}(0)$, the program cannot enter the if loop.

step 3

print. $n = 0$ as the program initialized at the beginning.

c) The "foo" function will be called again and again until the stack corrupts the memory region of program and static variables

(your own)

9.2 a) Here you are required to give assumption of keeping

A, B, and C atomic.

Version 1: Disable interrupt before A, B, C
Enable interrupt after A, B, C

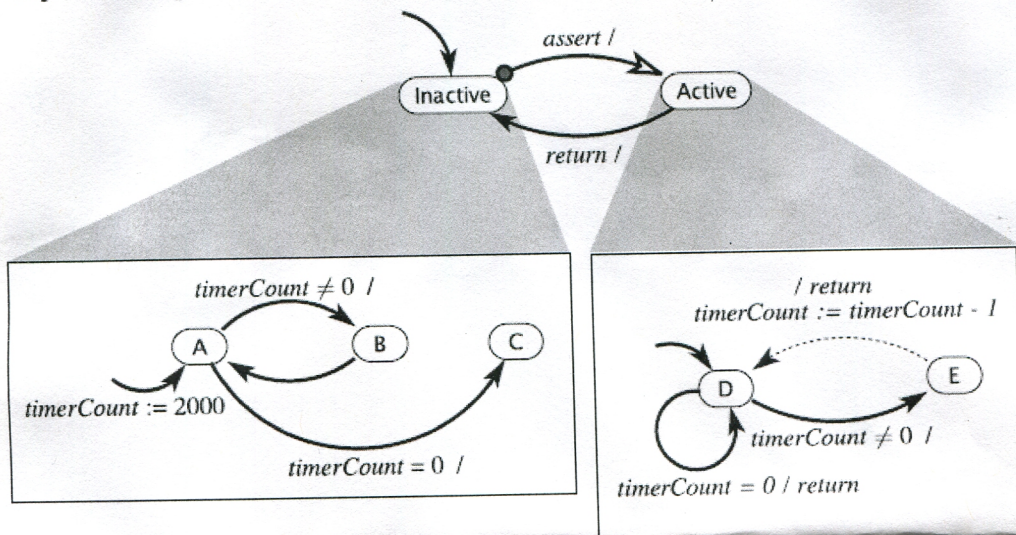
→ ISR, Many of you miss this one.

Version 2: Disable interrupt before A, B
Enable interrupt after A, B

Besides, function $\text{foo}()$ doesn't write or read variable timer_count .

b) You can have your own FSM, but it should be non-deterministic.
 Some of you referenced Fig 9.6, 9.7 in the textbook. The figures are right, but it doesn't guarantee the "Deterministic" characteristic that refers to the uniqueness of the computation according to the given input.

variables: timerCount: uint
 input: assert: pure, return: pure
 output: return: pure



- c)
- ① Non-deterministic.
 - ② Uncertain times.
 - ③ No, the program may enter C zero time or many times. Besides, a non-deterministic finite state machine is not able to provide an unique computational process as the programmers intend to.