Indian Institute of Information Technology, Sri City, Chittoor

Name of the Exam: Cyber-Physical Embedded Systems Duration: 2 hours Max. Marks: 100 (5 * 20)

Roll No.:_____Room No.: _____Seat No.: _____

Name: _____Invigilator's Signature: _____

Instructions:

1. This is a closed book examination.
2. If there are some values not given in this question paper, kindly assume standard values and state your assumption in the answer sheet. If you don't know the standard values use variable names and derive solutions with unknown variables.
3. Answer any five questions out of the 7 questions. Each question carries 20 marks (5* 20 = 100).

1.

A **tuning fork**, shown in Figure 2.5, consists of a metal finger (called a **tine**) that is displaced by striking it with a hammer. After being displaced, it vibrates. If the tine has no friction, it will vibrate forever. We can denote the displacement of the tine after being struck at time zero as a function $y: \mathbb{R}_+ \to \mathbb{R}$. If we assume that the initial displacement introduced by the hammer is one unit, then using our knowledge of physics we can determine that for all $t \in \mathbb{R}_+$, the displacement satisfies the differential equation

$$\ddot{y}(t) = -\omega_0^2 y(t)$$

where ω_0^2 is a constant that depends on the mass and stiffness of the tine, and where $\ddot{y}(t)$ denotes the second derivative with respect to time of y. It is easy to verify that y given by

$$\forall\, t \in \mathbb{R}_+, \quad y(t) = \cos(\omega_0 t)$$

is a solution to the differential equation (just take its second derivative). Thus, the displacement of the tuning fork is sinusoidal. If we choose materials for the tuning fork so that $\omega_0 = 2\pi \times 440$ radians/second, then the tuning fork will produce the tone of A-440 on the musical scale.

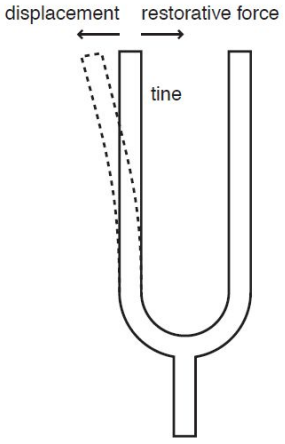


Figure 2.5: A tuning fork.

(a) Is $y(t) = \cos(\omega_0 t)$ the only solution? If not, give some others.

(b) Assuming the solution is $y(t) = \cos(\omega_0 t)$, what is the initial displacement?

(c) Construct a model of the tuning fork that produces y as an output using generic actors like Integrator, adder, scaler, or similarly simple actors. Treat the initial displacement as a parameter. Carefully label your diagram.

2.

Consider a rotating robot where you can control the angular velocity around a fixed axis.

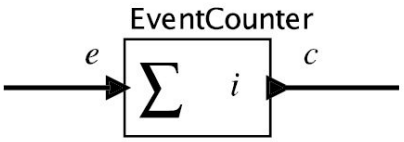
(a) Model this as a system where the input is angular velocity $\dot{\theta}$ and the output is angle θ. Give your model as an equation relating the input and output as functions of time.

(b) Is this model BIBO stable?

(c) Design a proportional controller to set the robot onto a desired angle. That is, assume that the initial angle is $\theta(0) = 0$, and let the desired angle be $\psi(t) = au(t)$, where u is the unit step function. Find the actual angle as a function of time and the proportional controller feedback gain K. What is your output at $t = 0$? What does it approach as t gets large?

3.

Consider an event counter that is a simplified version of the counter in Section 3.1. It has an icon like this:



This actor starts with state i and upon arrival of an event at the input, increments the state and sends the new value to the output. Thus, e is a pure signal, and c has the form $c\colon \mathbb{R} \to \{absent\} \cup \mathbb{N}$, assuming $i \in \mathbb{N}$. Suppose you are to use such an event counter in a weather station to count the number of times that a temperature rises above some threshold. Your task in this exercise is to generate a reasonable input signal e for the event counter. You will create several versions. For all versions, you will design a state machine whose input is a signal $\tau\colon \mathbb{R} \to \{absent\} \cup \mathbb{Z}$ that gives the current temperature (in degrees centigrade) once per hour. The output $e\colon \mathbb{R} \to \{absent, present\}$ will be a pure signal that goes to an event counter.

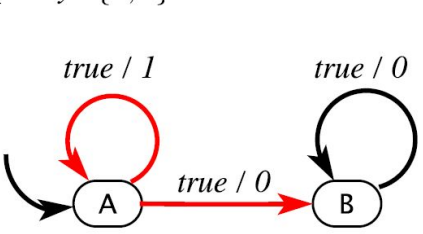
(a) For the first version, your state machine should simply produce a *present* output whenever the input is *present* and greater than 38 degrees. Otherwise, the output should be absent.

(b) For the second version, your state machine should have hysteresis. Specifically, it should produce a *present* output the first time the input is greater than 38 degrees, and subsequently, it should produce a *present* output anytime the input is greater than 38 degrees but has dropped below 36 degrees since the last time a *present* output was produced.

(c) For the third version, your state machine should implement the same hysteresis as in part (b), but also produce a *present* output at most once per day.

4.

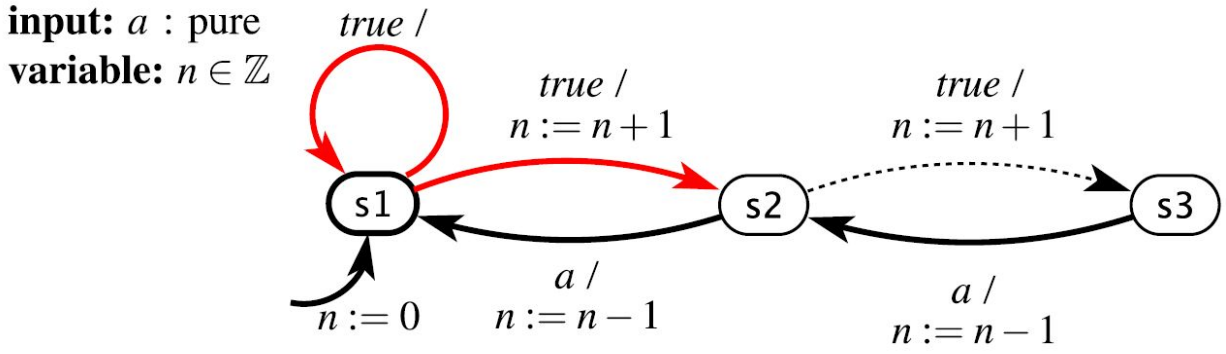
Consider the following state machine:



Determine whether the following statement is true or false, and give a supporting argument:

> The output will eventually be a constant 0, or it will eventually be a constant 1. That is, for some $n \in \mathbb{N}$, after the n-th reaction, either the output will be 0 in every subsequent reaction, or it will be 1 in every subsequent reaction.

How many reachable states does the following state machine have?



5.

Consider the following instruction, discussed in Example 8.6:

```
MAC *AR2+, *AR3+, A
```

Suppose the processor has three ALUs, one for each arithmetic operation on the addresses contained in registers AR2 and AR3 and one to perform the addition in the MAC multiply-accumulate instruction. Assume these ALUs each require one clock cycle to execute. Assume that a multiplier also requires one clock cycle to execute. Assume further that the register bank supports two reads and two writes per cycle, and that the accumulator register A can be written separately and takes no time to write. Give a reservation table showing the execution of a sequence of such instructions.

Example 8.6: The Texas Instruments TMS320c54x family of DSP processors is intended to be used in power-constrained embedded applications that demand high signal processing performance, such as wireless communication systems and personal digital assistants (**PDA**s). The inner loop of an FIR computation (8.1) is

```
1 RPT numberOfTaps - 1
2 MAC *AR2+, *AR3+, A
```

The first instruction illustrates the **zero-overhead loop**s commonly found in DSPs. The instruction that comes after it will execute a number of times equal to one plus the argument of the `RPT` instruction. The `MAC` instruction is a **multiply-accumulate instruction**, also prevalent in DSP architectures. It has three arguments specifying the following calculation,

$$a := a + x * y \ ,$$

where a is the contents of an accumulator register named `A`, and x and y are values found in memory. The addresses of these values are contained by auxiliary registers `AR2` and `AR3`. These registers are incremented automatically after the access. Moreover, these registers can be set up to implement circular buffers, as described in the box on page 221. The c54x processor includes a section of on-chip memory that supports two accesses in a single cycle, and as long as the addresses refer to this section of the memory, the `MAC` instruction will execute in a single cycle. Thus, each cycle, the processor performs two memory fetches, one multiplication, one ordinary addition, and two (possibly modulo) address increments. All DSPs have similar capabilities.

6.

1. Consider the function `compute_variance` listed below, which computes the variance of integer numbers stored in the array `data`.

```
1  int data[N];
2
3  int compute_variance() {
4    int sum1 = 0, sum2 = 0, result;
5    int i;
6
7    for(i=0; i < N; i++) {
8      sum1 += data[i];
9    }
10   sum1 /= N;
11
12   for(i=0; i < N; i++) {
13     sum2 += data[i] * data[i];
14   }
15   sum2 /= N;
16
17   result = (sum2 - sum1*sum1);
18
19   return result;
20 }
```

Suppose this program is executing on a 32-bit processor with a direct-mapped cache with parameters $(m, S, E, B) = (32, 8, 1, 8)$. We make the following additional assumptions:

- An `int` is 4 bytes wide.
- `sum1`, `sum2`, `result`, and `i` are all stored in registers.
- `data` is stored in memory starting at address `0x0`.

Answer the following questions:

(a) Consider the case where `N` is 16. How many cache misses will there be?

(b) Now suppose that `N` is 32. Recompute the number of cache misses.

(c) Now consider executing for `N` = 16 on a 2-way set-associative cache with parameters $(m, S, E, B) = (32, 8, 2, 4)$. In other words, the block size is halved, while there are two cache lines per set. How many cache misses would the code suffer?

7. *Scheduling (20 marks):* For the below set of sub-questions, assume period equals deadline; assume all instances of all tasks arrive at time instance zero; assume the processes are periodic; execution time is constant for each instance of the process.

Process	**Execution Time**	**Period**
1	1	4
2	2	6
3	3	8

1. Compute the processor utilization under Rate Monotonic Scheduling for the process, execution time and period as per given in the above table *(2 marks)*,
2. For the Table given above, compute the least upper bound on the processor utilization under Rate Monotonic Scheduling *(2 marks)*,
3. Comment on the difference between the actual and the least upper bound processor utilization for the Table given above *(4 marks)*,
4. Construct the schedule under Rate Monotonic Scheduling for the above table *(12 marks)*.
